AD-A037 564 HARRY DIAMOND LABS ADELPHI MD F/G 9/2
GRAPHELP USER'S MANUAL: VERSION 2.(U)
MAR 77 S CHOY
UNCLASSIFIED HDL-TM-77-2 NL

| OF |
AD
A037564

END

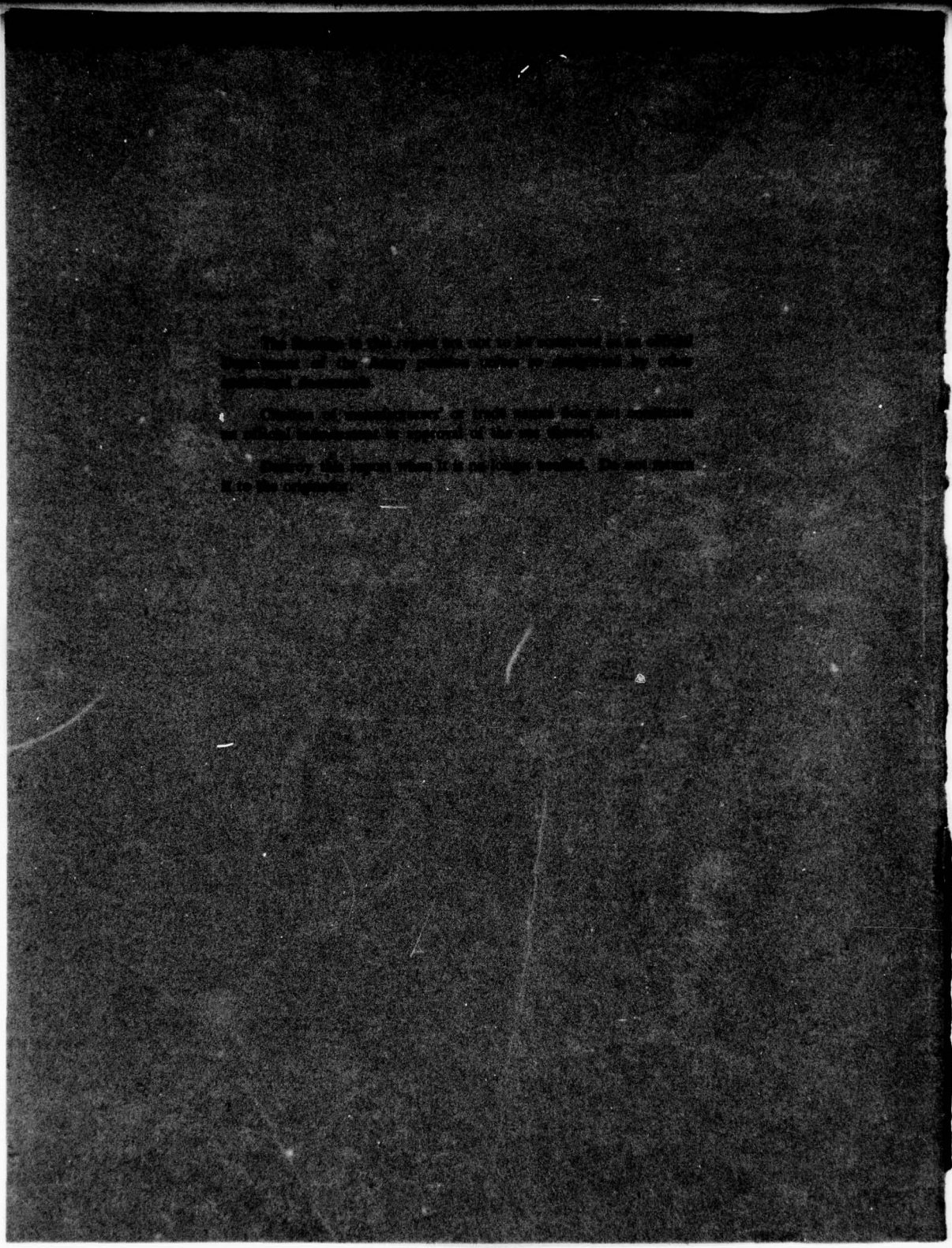
DATE
FILMED
4-77

HDL-TM-77-2

GRAPHELP USER'S MANUAL: Version 2

March 1977

U.S. Army Material Development
and Readiness Command
HARRY DIAMOND LABORATORIES
Adelphi, Maryland 20783



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

(14) 1. REPORT NUMBER: HDL-TM-77-2

2. GOVT ACCESSION NO.

(9) 3. RECIPIENT'S CATALOG NUMBER

(6) 4. TITLE (and Subtitle): GRAPHELP User's Manual: Version 2.

5. TYPE OF REPORT & PERIOD COVERED: Technical Memorandum

6. PERFORMING ORG. REPORT NUMBER

(10) 7. AUTHOR(s): Steven Choy

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Harry Diamond Laboratories
2800 Powder Mill Road
Adelphi, MD 20783

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
US Army Materiel Development & Readiness Command
Alexandria, VA 22333

(11) 12. REPORT DATE: March 1977

13. NUMBER OF PAGES: 64

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

(12) 58 p.

15. SECURITY CLASS. (of this report): Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

HDL Project: 898C81
DRCMS Code: 36AA.5000

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Interactive graphics
Imlac PDS-4
Tektronix
Computer graphics
Graphics programming

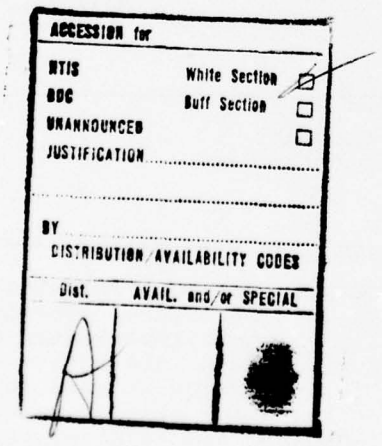
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An interactive-graphics, minicomputer-based, FORTRAN IV, software package called GRAPHELP has been developed. The current GRAPHELP software package has working versions implemented on three different minicomputer systems providing interactive graphics support for all Tektronix 401X graphics storage tube terminals and the Imlac PDS-4 refresh graphics display system. The software package provides both absolute and relative vectors → next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

 UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

cont.
of four varying line textures, user definable scaling, windowing, clipping, terminal transparency, and 128 nested subpicture display files for refresh graphics. In addition, routines are provided for interactive graphics crosshair input, interactive screen erase control, and selective erase to aid users in building their own interactive graphics display programs with GRAPHELP as the foundation. For those applications that are oriented toward data plotting rather than picture drawing, axes and scaling routines for both linear and logarithmic data are provided, along with alphabetic and numeric symbol output. This document both describes the software capabilities and provides instructions on its use.



CONTENTS

CONTENTS (Cont'd)

FIGURES

1. INTRODUCTION

GRAPHELP is a set of FORTRAN IV callable routines to represent data in pictorial form both interactively and passively on a cathode ray tube (CRT) graphic display. Currently, output drivers exist for the Imlac PDS-4 refresh display system and the entire family of Tektronix 401X Direct View Storage Tube (DVST) displays. Included is support for the enhanced Graphics mode for the Tektronix 4014 and 4015 high-resolution display terminals. Future drivers will be available for daisy-wheel plotting devices and the Interdata carousel wheel printer.

The programs are written entirely in FORTRAN and assume that the computer is a 16-bit, two's compliment machine with ASCII character formats. In addition, the following utility routines must be available in the FORTRAN library: a logical SHIFT, a logical OR, a logical AND, and an arbitrary eight-bit input and output to the serial communication lines of the computer. These routines have been isolated at the ends of the program for easy installation on new systems.

This manual (version 2) is a user's guide to GRAPHELP as it currently exists on the PRIME 400 DOS/VM and PDP 11/45 RSX-11D systems. Differences between version 1 and version 2 are summarized in appendix A. Instructions for using GRAPHELP on these two systems are given in appendix B.

2. OVERVIEW

The system is written on two levels of programming. At the lower level is a set of graphic primitives to draw lines, position the beam, erase the screen, output text, draw circles, draw arcs, and do a number of miscellaneous functions to give the user control over the state of the display. Beam positioning and line drawing are expressed in a standard relative unit of length that has been initially set at approximately 1 in. The unit of length can be changed by the user at any time. Once the unit of length is established, all vectors are drawn in relative scale to that length. Facilities are provided to interrogate the position of the cross-hair cursor and translate the raster units position into inches or to whatever the current unit of length in effect is at that time.

At the higher level, the user establishes his own logical unit system for his particular application. Essentially, an image space is established by calling scaling routines in two dimensions and translation coordinate in inches to position the origin of the image space on the screen. Routines are provided to perform automatic scaling both linearly and logarithmically. Once the image space or region is

defined, a user can draw in the logical coordinate system of the currently defined region. Up to seven regions can optionally be saved and invoked at any time. In addition, there are aids to draw x-y axes, linear grids, and plot data lines from x-y coordinate arrays. At this level, the user does not have to worry about the lower-level graphics being performed to implement the higher-level, two-dimensional graphics. The user can perform all his graphic routines in his own coordinate system without having to worry about scaling transformations, vector overflow on the screen, or any of the other common problems associated with graphic programming. Data plotting subroutines are provided for both incremental and point-by-point plotting. Options to set up multiple windows on the screen (up to seven) also are available. At the same time, the more ambitious programmer can decorate and customize his display to suit his application by calling other lower-level routines. This is the level of graphics programming that is the easiest and most convenient to use for most applications.

Because the graphic capabilities on the Imlac PDS-4 far exceed those functions available on the Tektronix storage-tube displays, many of the subroutines available on the Imlac PDS-4 cannot be used on the Tektronix tubes. Most of these routines are used with subpicture manipulation and Imlac PDS-4 hardware functions. However, if the Imlac routines are used (e.g., blinks and circle generator), the user program can still run on the Tektronix displays, except that these routines take no action. It is advised that subpictures not be used if CRT device independence is desired. For the sake of discussion, the user's logical coordinate space is called the "virtual" space in this manual.

3. GRAPHIC PRIMITIVES

The graphic primitives are the heart of the graphics routines. At this level, almost any type of display is available, limited only by one's imagination and the display hardware. The primitives are classified into four groups of routines: control functions, drawing routines, subpicture control, and miscellaneous functions. The control functions initialize graphics programs, enter and exit GRAPHICS mode, and erase the screen. Drawing routines provide a means to position the beam (move to a location) and draw lines, circles, arcs, and text with varying line types and beam intensity.

Subpicture control applies only to the Imlac PDS-4. It allows programs to present pictures in dynamic (animated) forms. Also, it allows multiple copies of a single image to be displayed with less data storage and less data communications transfer. The miscellaneous functions include utilities to perform functions not directly related to changing the display, but enhancing the interactive element between the person and the computer through the display.

Appendix C summarizes these routines and defines the calling parameters. Refer to this appendix in conjunction with the following descriptions of the routines.

3.1 Control Functions

All programs using GRAPHELP begin with a single call to SETPDQ. This is the initialization routine. It identifies the terminal, sets up all graphic routines, and initializes all internal parameters. Without this call, all graphic routines are invalid.

SETPDQ	initialization of GRAPHELP
ENTGRA	enter GRAPHICS mode
EXITGR	exit GRAPHICS mode
ERASEQ	erase screen
ERASEA	erase alpha display
ERSALL	erase screen, clear all subpictures
EOFXMT	exit graphics and inhibit all output to the display until a keyboard input
SCREEN	erase screen while in ALPHA mode
FACTOR(XFACT, YFACT)	set overall output size

A GRAPHELP system can be in either of two modes: GRAPHICS or ALPHA (not in GRAPHICS mode). To enter GRAPHICS mode, call ENTGRA. To exit GRAPHICS mode and enter ALPHA mode, call EXITGR. A program is considered to be initially in ALPHA mode until a call to ENTGRA is made. All FORTRAN read and write statements must be made while in ALPHA mode only. If this rule is violated, the results are unpredictable.

With the exception of IWAITQ and SCREEN, almost all GRAPHELP routines must be called while in GRAPHICS mode. Appendix D lists all routines that must be called in GRAPHICS mode.

Erasing the screen can be done in a number of different ways, depending on which terminal the program is being used. On Tektronix storage tube terminals, all erase commands erase the entire screen. On the Imlac PDS-4, output to ALPHA mode and GRAPHICS mode is controlled separately. Therefore, the displays can be erased separately. ERASEQ erases both displays. ERASE erases the ALPHA display. ERSALL erases both displays and all subpicture definitions. SCREEN erases both displays, but is called only while in ALPHA mode. EOFXMT is a very special EXITGR. It exits graphics and then inhibits all further output to the screen until a keyboard input is made. This inhibition prevents unwanted operating system messages from appearing in the main display on program completion.

FACTOR sets the length of a unit of measurement. It acts like an overall scale factor of all lines put on the display. Initially, FACTOR is assumed to be (1.,1.), which means that one unit is equivalent to about 1 in. For this reason, the inch is used as the standard unit of length in this manual for describing line-length parameters. This is only a relative unit of length. If FACTOR is called with 0.5, then this unit of measurement becomes 0.5 in.

Example 1

```
CALL SETPDQ              initialize all graphics, including factor to 1.
CALL SCREEN              erase screen
CALL ENTGRA              enter graphics
CALL RELVEC(1.0,0.0,1)   draw 1-in. vector
CALL FACTOR(2.0,2.0)     double factor size
CALL RELVEC(1.0,0.0,1)   draw 1-in. vector
CALL EXITGR              exit GRAPHICS mode
STOP
END
```

Although both calls to RELVEC are 1 in. long, the second call actually draws a line twice as long as the first.

The most obvious applicaton for this is to use FACTOR to reduce or enlarge the entire size of the display without having to change all the drawing commands.

Example 2

```
        CALL SETPDQ
5       READ(1,10) ISIZE            input factor size
10      FORMAT(I2)
        SIZE=FLOAT(ISIZE)
        CALL FACTOR(SIZE,SIZE)
        CALL SCREEN                 erase screen
        CALL PICTUR                 call arbitrary picture
        GO TO 5
        STOP
        END
```

In example 2, it is assumed that the PICTUR subroutine draws some arbitrary picture with no calls to FACTOR, and it begins and ends with calls to ENTGRA and EXITGR. The size of the picture can be changed in each successive loop of the read statement into ISIZE.

3.2 Drawing Routines

The drawing routines are the heart of all display generation in GRAPHELP. These are the routines that draw the points, lines, arcs, circles, and text. In addition, intensity control is available along with 90-deg rotations on the Imlac PDS-4. These routines may be called only while in GRAPHICS mode.

ABSVEC(XAB,YAB,LINETP)	draw absolutely to XAB,YAB
RELVEC(X,Y,LINETP)	draw relative to current location
CIRQLE(RADIUS,LINETP)	draw circle (Imlac hardware only)
ARC(XCR,YCR,ANGL,LINETP)	draw arc (Imlac hardware only)
SYMBOQ(NCHAR,ITEXT,NSIZE)	draw text in graphics
ROTA90(N90)	rotate in 90-deg increments (Imlac only)
INTENS(IBRITE)	change screen level intensity (Imlac only)
BLINKQ	toggle blink on and off (Imlac only)

Coordinates or vector components are expressed in inches. Rotation is expressed in terms of the number of 90-deg rotations. Brightness is a level from 1 to 16, where 1 is the dimmest. Angles in the circle and arc routines are specified in radians, and character size is a level from one to seven, where one is the smallest.

Depending on the terminal, several different line types are available. If a line type is called that is not available on the output terminal, the resulting line is drawn as a solid line. Lines can be drawn with the following types: invisible, solid, dot, dash, and dot-dash. See appendix B for the calling value of each line type.

Two methods of line drawing are available in GRAPHELP: an absolute vector and a relative vector. An absolute vector is invoked by a call to ABSVEC, and a relative vector is invoked by a call to RELVEC.

Relative lines have the advantage of being position independent, but have the disadvantage of the user's having to know the length of each line. Absolute lines work inversely. They have the advantage of the user's not having to know the length of each line, but have the disadvantage of being position dependent. The absolute vectors are good for drawing from point to point.

Example 3

Draw a dotted triangle at coordinates (1,1),(1,2),(4,4):

```
CALL SETPDQ
CALL SCREEN
CALL ENTGRA
CALL ABSVEC(1.,1.,0)     position, beam off to first point
```

```
      CALL ABSVEC(1.,2.,2)     draw dotted beam
      CALL ABSVEC(4.,4.,2)     draw dotted beam
      CALL ABSVEC(1.,1.,2)     close triangle
      CALL EXITGR
      STOP
      END
```

Example 4

Draw a dot-dash rectangle 1 × 2 beginning at (3,3):

```
      CALL SETPDQ
      CALL SCREEN
      CALL ENTGRA
      CALL ABSVEC(3.,3.,0)
      CALL RELVEC(2.,0.,4)
      CALL RELVEC(0.,1.,4)
      CALL RELVEC(-2.,0.,4)
      CALL RELVEC(0.,-1.,4)
      CALL EXITGR
      STOP
      END
```

By using RELVEC and FACTOR, a subroutine can be written to produce a position-independent box of varying size and with variable line types.

Example 5

Draw position-variable box:

```
      SUBROUTINE VBOX(XFAC,YFAC,LINETP)
      CALL FACTOR(XFAC,YFAC,LINETP)
      CALL RELVEC(1.,0.,LINETP)
      CALL RELVEC(0.,1.,LINETP)
      CALL RELVEC(-1.,0.,LINETP)
      CALL RELVEC(0.,-1.,LINEPT)
      CALL FACTOR(1.,1.)
      RETURN
      END
```

Now, different boxes can be drawn anywhere on the screen by calling VBOX, because VBOX is completely position independent.

Example 6

```
      CALL SETPDQ
      CALL SCREEN
```

```
      CALL ENTGRA
      CALL ABSVEC(1.,1.,0)
      CALL VBOX(1.,1.,2)
      CALL ABSVEC(4.,4.,0)
      CALL VBOX(2.,1.,3)
      CALL EXITGR
      STOP
      END
```

Text is generated in GRAPHICS mode by positioning the beam to the beginning point of the text (lower left corner) and then calling SYMBOQ. Character sizes are dependent on the output terminal. The Tektronix 4010, 4012, and 4013 have only one size. The Imlac PDS-4 has seven sizes of character. The Tektronix 4014 and 4015 have four sizes. Consequently, on the Tektronix 4014, 1 and 2 get mapped to the same size; and similarly, 3 and 4, 5 and 6, and 7 and 0 all get mapped to the same size. If a negative value is used for the number of characters, then SYMBOQ draws the characters vertically with one under the other. The absolute value of this number if used as the character count.

Output text in the form of an array of Hollerith text is passed to SYMBOQ which outputs the characters one at a time. If it is desired to display variables in specific formats, then the variables must be converted to Hollerith or ASCII form. This can be done by using ENCODE; then, the Hollerith string is passed normally.

Example 7

```
      DIMENSION ITEXT(25)
      CALL SETPDQ
      CALL SCREEN
      CALL ENTGRA
      CALL ABSVEC(4.0,4.0,0)
      J=25
      K=76
      CALL ENCODE(21,10,ITEXT)J,K       convert J and K to ASCII via
                                              format 10
10    FORMAT ('THIS IS J ',I2,' AND K ',I2)
      CALL SYMBOQ(21,ITEXT,5)
      CALL ABSVEC(3.0,3.0,0)            position beam
      CALL SYMBOQ(-28,'THIS IS VERTICAL TEXT SIZE 2',2)
      CALL EXITGR
      STOP
      END
```

In example 7, the variables J and K are displayed by using format 10 at location (4,4) with a size-5 character. The output would look like this:

```
THIS IS J 25 AND K 76
```

The second call to SYMBOQ would output the message contained in Hollerith form (between the quotation marks) at location (3,3) and draw it size 2 and vertically. For programs with many messages and titles, it is usually easier to store the text in arrays with data statements (example 8).

Example 8

```
      DIMENSION M1(15),M2(15),M3(15)
C
C SET UP MESSAGE ARRAYS
C
      DATA M1 /17HTHIS IS MESSAGE 1/
      DATA M2 /17HTHIS IS MESSAGE 2/
      DATA M3 /17HTHIS IS MESSAGE 3/
      CALL SETPDQ
      CALL SCREEN
      CALL ABSVEC(1.,1.,0)
      CALL SYMBOQ(17,M1,3)              message 1
      CALL ABSVEC(2.,2.,0)
      CALL SYMBOQ(17,M2,3)              message 2
      CALL ABSVEC(3.,3.,0)
      CALL SYMBOQ(17,M3,3)              message 3
      CALL EXITGR
      STOP
      END
```

CIRQLE and ARC routines run only on the Imlac PDS-4 with the hardware circle generator. *CIRQLE assumes that the beam* is positioned at the center of the circle at calling time. CIRQLE ends with the beam at the center of the circle at exit time. Line types are invisible, solid, or dotted.

ARC assumes that the beam is at the starting point of the arc. XCR and YCR designate the center of the arc relative to the starting point. Angle is expressed in radians where a negative angle is clockwise. ROTA90, INTENS, and BLINKQ all work only on the Imlac PDS-4. BLINKQ toggles the blink mode on and off. Intensity varies the beam brightness, and ROTA90 rotates all following lines in 90-deg increments. A call to ROTA90 with a zero resets all rotation parameters. Rotations are composite functions that accumulate on each call to ROTA90.

3.3 Subpicture Control

Subpictures are used only on the Imlac PDS-4. Part 5 of this manual is devoted to the subpictures.

OPENQ(NAME)	define new subpicture (redefine old one)
OPPICT(IPIC)	same as OPENQ, but identify by number
CLOSEQ	end define subpicture mode
ADDON(NAME)	append to old subpicture
ADPICT(IPIC)	same as ADDON, but identify by number
ADDOFF	end append subpicture mode
DIQPLA(NAME)	display already defined subpicture
DSPICT(IPIC)	same as DIQPLA, but identify by number
ERSPIC(NAME)	erase already defined subpicture
ERPICT(IPIC)	same as ERSPIC, but identify by number
EMPTYP(NAME)	empty contents of subpicture
SERCH(NAME,INDEX)	look up subpicture index
ATACHP(NAME)	attach subpicture to cross hair by name
ATPICT(IPIC)	attach subpicture to cross hair
DEPICT(IPIC)	detach subpicture from cross hair

3.4 Miscellaneous Functions

Miscellaneous functions include those routines that do not directly affect the state of the display. In addition, utilities are provided to enhance the interaction between the computer and the user through the display terminal. In both the Imlac PDS-4 and the Tektronix storage-tube terminals, a cross-hair cursor is used to relate coordinate positions to the computer.

WILDCR(NWILD)	set overall output size
PROMPT(ISWIT)	interrogate Imlac data register
INQUIR(XRAS,YRAS,ICHAR)	turn on and get cross-hair position
ENQUIR(IXRAS,IYRAS,ICHAR)	get cross-hair position in integer rasters
DELAQ(NSEC)	PRIME delay in seconds
DELAY(NULLS)	output nulls for delay
RSINCH(XRAS,YRAS,XAB,YAB)	convert rasters to inches
INVIRT(XAB,YAB,XS,YS,ITYPE)	convert inches to virtual
RSVIRT(XRAS,YRAS,XS,YS,ITYPE	convert raster to virtual
FUNCTION IWAITQ(IOPCO)	prompt, wait, and erase
HDCOPY(ISWIT)	make hard copy of the GRAPHICS file
SETPAG(XAB)	set CalComp plotter end of page
SETTAB(IPREC,LINETP,IUNITS, ITYPE,IDISA)	set PDS-4 tablet and INQUIR mode for all terminals

INQUIR enables the cross-hair cursor on the screen. The position of the cross hair can be adjusted by the thumb wheels on the Tektronix and by the function arrows on the Imlac PDS-4. INQUIR waits for a key to be struck on the keyboard and then disables the cursor. The absolute raster unit position is returned in XRAS,YRAS. The key

struck on the keyboard is returned in ICHAR in an A1 format or a single left-justified character blank filled. It is important to remember that the Tektronix terminals must be strapped to send the automatic carriage return on ENQ mode. An optional feature of INQUIR is the capability to return the coordinate values in one of the two other coordinate systems. By calling SETTAB with the appropriate arguments (app C), the x,y coordinate pair may be returned in raster units, inches, or the user's own virtual coordinate system. Finally, to allow maximum efficiency and flexibility, ENQUIR is provided as an alternative to INQUIR to return the cross-hair position in integer raster unit format. This alternative is for the programmer who does not wish to deal with real floating point numbers.

In many cases, the raster-unit coordinate position is not useful. RSINCH can be used to translate the raster coordinate positions to the current factor unit of length (inches). Also available is INVIRT to translate inches to virtual coordinates and RSVIRT to translate raster units to virtual coordinates. With the combination of these three routines, mappings from any coordinate system to the other are immediately available.

In many cases, it is desirable to suspend a program temporarily on the completion of display generation. This suspension allows the user to look at the display for undefined lengths of time, make a hard copy, or make decisions whether to continue the program or to take some other action. The IWAITQ function has been supplied to perform such a function. IWAITQ is not a subroutine call, but a function call. It can be called only while in ALPHA mode. It has six different operation (op) codes to perform different functions. The returned function value is a character typed on the keyboard in A1 format (1H1).

IWAITQ has three basic functions, depending on the op code. It prints a message on the terminal indicating that it is to wait for input; it waits for input from the terminal; it erases the screen after the input. Because the nature of the interaction is different between refresh and storage-tube terminals, some op codes take one action on the Tektronix and a different action on the Imlac PDS-4.

See appendix B for a listing of all possible op codes.

Example 9

```
    DATA ICMD/1HN/
    CALL SETPDQ                          initialize
    I=0
    CALL SCREEN                          erase screen
10  I=I + 1
    CALL ENTGRA
```

```
CALL PICTUR(I)
CALL EXITGR
IF (IWAITQ(2).EQ.ICMD) GO TO 10      wait for character
STOP
END
```

Example 9 assumes that PICTUR(I) draws a picture depending on the value of I. IWAITQ suppresses the INPUT message, but waits for keyboard input and then erases the screen. If an N is typed, then a new picture is drawn, or else the program halts. This halt gives the user time to look at the picture between changes and to decide whether to continue.

Delay functions are furnished in two forms. DELAY outputs nulls to the terminal, and DELAQ loops in the central processing unit (CPU) for the requested number of seconds. DELAQ is only an approximate waiting time, depending on how heavily the system is loaded.

Except for HDCOPY and SETTAB, the remaining routines apply to the Imlac PDS-4 only.

PROMPT gives the user the capability to interrogate the Imlac PDS-4 switch registers. Only the right three bits are read. PROMPT returns a value from zero to seven. SETPAG sets the terminating x address in absolute inches for the Calcomp drum plotter. Any real number (plus or minus) is legal.

HDCOPY tells the graphic terminal to put out a hard copy. The ISWIT parameter does nothing on the Tektronix terminals, but returns the prompt character in the switch register for the Imlac PDS-4.

SETTAB is used to set which coordinate system INQUIR should use to return its values. In addition, on the Imlac PDS-4, SETTAB is used to set various data tablet parameters. These parameters are discussed in section 5 of this manual. For INQUIR mode, IUNITS and ITYPE indicate the coordinate system. If SETTAB is not used, then INQUIR defaults to the raster unit coordinate system. See section 5 for more details on SETTAB.

WILDCR sends out the desired wild-card number for the Imlac PDS-4. The following is a list of wild-card numbers available:

1 erase ALPHA display list
2 toggle ALPHA list visibility
3 toggle ALPHA cursor visibility
4 erase GRAPHICS
5 toggle GRAPHICS list visibility

```
 6   toggle GRAPHICS cursor visibility
 7   erase ALPHA and GRAPHICS list
 8   toggle cross hair
 9   start paper tape reader to display
10   local mode
11   on-line mode
12   start/stop paper tape punch
13   paper tape off
14   transparent angle
15   error recovery
16   selective subpicture erase
17   change ALPHA scale in graphics
18   read switch reg and send ASCII
19   blink control
20   transmit cross hair position
21   arbitrary binary input
22   toggle header visibility
23   change dash mode
24   set Calcomp end of page position
25   attach subpicture to cross hair
26   detach subpicture from cross hair
27   not used
28   turn on tablet trace mode
29   enable tablet trace line
30   disable tablet trace line
31   set tablet precision
```

This concludes the descriptions of the graphics primitives. Refer to appendix B for a summary of all the routines.

It is important that SCREEN and IWAITQ be used only while in GRAPHICS mode; otherwise, results are unpredictable.

4. TWO-DIMENSIONAL PLOTTING AIDS

The two-dimensional plotting routines allow a user to define his image space in which he can draw in his coordinate system. The user can specify the origin and the scaling transformations in both x and y directions along with an overall factor size. These image space parameters may be optionally stored into a space called a region. Up to seven image spaces or regions can be stored. Using the appropriate routines, the user can now specify lines, axes, grids, and data plots in his own dimensions.

At this level, all vectors, both relative and absolute, can be specified in the user's dimensions. At the user's option, these vectors may be clipped at either a previously defined window boundary or at a

previously defaulted window boundary established by the earlier use of an automatic scaling routine. With the combination of this clipping facility and the user's definable scaling and origin translation, a software zooming capability is easily available.

4.1 Scaling

To use these routines, the user must define his origin and scaling parameters before any drawing can be done. To save the current image space, a STOREG must be called to save the region. At any point later in time, the region may be recalled after it has been saved with a REGION call. Setting the origin is accomplished by calling INIT(XIN,YIN). This defines the origin point for the current image space at XIN,YIN. Scaling may be accomplished by calling one of five routines. Both x and y directions must be scaled. The routines are as follows:

SCALE(ARRAY,NPTS,AXLEN,INC,IWHO)	linear scaling
QCALE(AMIN,AMAX,AXLEN,SF,VLO,IWHO)	linear scaling
LOGQCA(ARRAY,NPTS,AXLEN,INC,IWHO)	log scaling
LOGSCA(AMIN,AMAX,AXLEN,SF,VLO,IWHO)	log scaling
YOURSC(SF,VLO,IWHO)	set user's own scale
RESCA(ARRAY,NPTS,AXLEN,IWHO,ITYPE)	rescale

AXLEN is the length of the axis (page length) to be scaled in inches. NPTS is the number of points in ARRAY that contains the data to scale. INC is the rate at which the data are sampled (usually 1). IWHO is the axis specifier where IWHO=0 indicates the x-axis, and any other value is the y-axis.

Using SCALE or LOGQCA, the user passes an array of data points to be scaled where SCALE performs linear scaling and LOQSCA does logarithm scaling. Alternatively, QCALE and LOGSCA calculate scale factors given a minimum and a maximum value in AMIN and AMAX. In addition, the two scaling parameters are returned where SF is the scale factor in units per inch and VLO is the calculated value for the origin. For log scaling, SF represents the scaling factor in log units per inch.

If the user does not wish to use the automatic scaling features, he may specify a self-computed scale factor and pass it to GRAPHELP using YOURSC. If the user desires to know what the current scale factor in effect is, then the route GETSCA can be called:

```
CALL GETSCA(SF,VLO,IWHO)
```

For multiple array scaling, RESCA is provided. This routine can be called only after one of the other scaling routines has been called. RESCA uses the old extreme points of the current scaling and calculates new ones from ARRAY and the old extreme points.

Once the scaling parameters and the origin location are established, these current image-space parameters can be saved in a list up to seven image spaces or regions. There are two region commands: one to save a region and one to recall a region.

```
STOREG(N)
REGION(N)
```

STOREG stores the current region, and REGION recalls a previously stored region where N is the region number identifier from one to seven.

4.2 Drawing

Once the image space is defined, vectors can be drawn as both absolute and relative in the current scaling units relative to the currently defined origin. All transformations are made automatically to allow the user to draw using his defined image space. Four line drawing routines parallelling ABSVEC and RELVEC are provided.

```
DRAW(XS,YS,IPEN,ITYPE)    draw absolute in current scaled units
RDRAW(XS,YS,IPEN,ITYPE)   draw relative in current scaled units
DRAWC(XS,YS,IPEN,ITYPE)   same as DRAW with clipping
RDRAWC(XS,YS,IPEN,ITYPE)  same as RDRAW with clipping
```

XS and YS are the coordinate points in the current scaling translation system, and ITYPE is the current type of scaling in effect. Appendix E has the full parameter descriptions.

Example 10

Set up two virtual spaces and draw in them

```
      SUBROUTINE BLOWUP(X,Y,N,B1,B2)
      DIMENSION X(1),Y(1)
      CALL SETPDQ
      CALL SCREEN

C
C  ESTABLISH REGION 1
      CALL INIT(1.,1.)
      CALL SCALE(X,N,4.,1,0)
      CALL SCALE(Y,N,4.,1,1)
      CALL STOREG(1)
C
C  ESTABLISH REGION 2
      CALL INIT(4.5.,4.5.)
```

```
         CALL QCALE(B1,B2,4.,SF,VLO,0)
         CALL QCALE(B1,B2,4.,SF,VLO,1)
         CALL STOREG(2)
C  DRAW IN THESE REGIONS
         CALL ENTGRA
         DO 10 I=1,N
C
C  DRAW IN SPECIFIED REGION
           DO 20 J=1,2
              CALL REGION(J)
              CALL DRAWC(X(I),Y(I),3,4)     position beam
              CALL DRAWC(X(I),Y(I),2,4)     draw point
20         CONTINUE
10       CONTINUE
         CALL EXITGR
         END
```

In example 10, subroutine BLOWUP takes data from arrays X and Y and draws a point-by-point plot in the two regions. B1 and B2 are minimum and maximum values of a square area that is blown up into REGION 2. The square is a small subset of REGION 1.

When values fall in this square, they are plotted in both REGION 1 and REGION 2. REGION 2 is scaled to blow up this area or look at it at higher magnification than REGION 1. Although identical DRAW commands are used for both plots, the virtual spaces are different.

Using DRAWC instead of DRAW forces the clipping on the window boundaries established by the automatic scaling QCALE. In some cases, the default window established by QCALE is not desirable. The window may be changed unconditionally at any time by the following call.

```
CALL SETWIN(XMN,YMN,XMX,YMX)
```

The parameters XMN and YMN represent the lower left corner of the window in the user coordinate system, and the values XMX and YMX represent the upper right corner of the clipping window in the user coordinate space.

If the user wishes to draw absolute vectors not transformed by scale, but still relative to the current origin, then a call to PLOT(XIN,YIN,IPEN) can be made. This routine draws in absolute inches translated by the defined origin. If IPEN is less than or equal to zero, then a new origin is defined by XIN,YIN.

If the data are linear in both X and Y, a grid can be drawn on the x-y-axes by calling GRID(XGRD,YGRD,XD,YD,IPEN,IREL). If IREL is equal to 1, then the grid is drawn entirely with relative vectors. This

action makes the grid position independent. XGRD,YGRD are the coordinates of the upper right corner in inches of the grid, and XD,YD are the distances between grid lines on the axis.

4.3 Axes Drawing

Axes support is provided by six routines. The axes are drawn at the origin. RXAXIS and RYAXIS are drawn only with relative vectors for position-independent coordinate graphs. The log axes are drawn in increments of full log cycles only.

```
XAXIS(LABEL,NCHAR,AXEN)        draw x linear axis
YAXIS(LABEL,NCHAR,AXLEN)       draw y linear axis
RXAXIS(LABEL,NCHAR,AXLEN)      draw x relative linear axis
RYAXIS(LABEL,NCHAR,AXLEN)      draw y relative linear axis
XLOGAX(LABEL,NCHAR,AXLEN)      draw x log axis
YLOGAX(LABEL,NCHAR,AXLEN)      draw y log axis
```

LABEL is the title to be printed on the axis. On the x-axis, the label is drawn left to right under the axis. On the y-axis, the label is drawn top to bottom to the left of the axis. NCHAR is the number of characters to print in the label. Care should be taken to put the origin point sufficiently in the screen to allow room to print the label. The number of digits that is displayed to the right of the decimal point has a default value of 2. This number can be changed by calling AXPREC(N), where N is a number greater than zero representing the desired number of digits. If used, AXPREC must be called before the axis call is made.

Two data plotting routines exist to plot x-y data for the user.

```
DATAQ(XARA,YARA,NPTS,INC,IPEN)          plot linear data only
DATLOG(XARA,YARA,NPTS,INC,IPEN,ITYPE)   plot data
```

Appendix D describes the calling parameters. DATLOG plots linear data, log-log data, or semilog data. DATAQ plots linear data only and is put there as a convenience for the user.

Annotating graphs can be performed by using SYMBOQ (sect. 3.2) or by using NUMBRQ. This is the number counterpart to SYMBOQ. It allows display of floating point numbers. The arguments are described in appendix D. This routine saves the user the trouble of converting numbers from floating point to ASCII and then calling SYMBOQ to display it. Only floating point numbers are available, but they can be displayed with no decimal portion.

Other small utilities exist for the user, including a run-time format calculator for floating point numbers and a minimum-maximum value

extractor for an array of numbers. The calling sequences also can be found in appendix D.

4.4 General Graph

In summary, to use the two-dimensional plotting package, one should follow the general procedure as follows:

a. Establish an origin point (leave room for axis labels).

b. Establish a scaling system in both x and y.

c. Draw axis and grids (always in GRAPHICS mode).

d. Plot data.

For the user that does not wish to bother with graphic protocols, but wants a quick output of a data plot, a routine called GENGRF is available. In this case, no calls to ERASEQ, SETPDQ, FACTOR, or anything else is necessary. In fact, all that is required is that the data be passed, and GENGRF can produce an aesthetic-looking x-y plot interactively on the terminal with automatic scaling, axes labels, and title.

In addition, GENGRF allows the user to interactively modify the format of the output to some small degree at run time. GENGRF plots only a linear graph currently. The call is GENGRF(X,Y,N), where X and Y are arrays of length N containing the data.

The two-dimensional plotting package has been written to accommodate two kinds of programmers. There is the easy, default, "do everything" version, and there is the more flexible version to perform specific types of display. This accommodation allows the user either to be satisfied with all the defaults and not worry or to go in and customize his program to generate a nicer display.

The typical format of a plotting program is shown in example 11.

Example 11

```
      DIMENSION X(N),Y(N)
      CALL SETPDQ                          set up GRAPHELP
      CALL SCREEN                          erase screen
      CALL INIT(2.,2.)                     initialize origin
      CALL SCALE(X,N,5.,1,0)               scale in x
      CALL SCALE(Y,N,5.,1,1)               scale in y
      CALL ENTGRA                          enter GRAPHICS mode
      CALL XAXIS('XAXIS',5,5.)             draw x axis
      CALL YAXIS('YAXIS',5,5.)             draw y axis
      CALL DATAQ(X,Y,N,1,4)                draw data plot
      CALL EXITGR                          exit GRAPHICS mode
      STOP
      END
```

Assuming that the data were in x and y, each array having N points, this program would produce a linear graph with the origin at 2.,2. and a data plot using a dotted line.

The user is not restricted to using DATAQ and DATLOG to draw his plots. DRAW and RDRAW can be used to plot the data point by point and, using RELVEC, one may form plotting symbols in subroutines to draw at some interval of data points.

Using the region routines, a user can define multiple regions on the screen and thereby divide the screen into different partitions. Each partition can have its own origin and scaling parameters.

By using RXAXIS, RYAXIS, RDRAW, and RDRAWC, plots may be put into subpictures and become position independent on the screen. This independence allows repositioning of an entire plot by simply repositioning a single subpicture without having to retransmit all the drawing data.

If particular values need to be displayed on the graph, then NUMBRQ can be used to display the floating point number. No format specification is necessary. SYMBOQ can be used to print identifying information using PLOT to position the text on the screen.

If the user decides that the entire plot is too big or small and does not wish to fuss with the scaling, then he can call FACTOR to enlarge or shrink the plot at the beginning. This call affects the entire output without changing the scaling. Section 6 of this manual contains a number of useful programming examples to demonstrate the different capabilities of GRAPHELP.

5. IMLAC SPECIAL FEATURES

The Imlac PDS-4 has some special features that are described here in detail. With these special features, which are not available on the Tektronix storage-tube graphics terminals, one may create a very complex interactive graphics system for particular applications. Besides the display enhancements of blink, variable intensity, and hardware 90-deg rotations, the Imlac PDS-4 offers the capability of defining 128 separately controlled display files called subpictures. In addition, there are alternative forms of graphic input devices such as the sonic data tablet with the Graphics Terminal System (GTS) cross hair. (GTS commands are summarized in app F.)

5.1 Subpicture Control

A subpicture is a collection of display commands stored in a file under a name that can be created, redefined, displayed, appended to, and deleted. Subpictures are implemented in a distributive processing method that uses the Imlac PDS-4 for storage and the host computer to generate the display commands.

OPENQ(NAME)	define new subpicture (redefine old one)
OPPICT(INDEX)	same as OPENQ (open by INDEX)
CLOSEQ	end define subpicture mode
ADDON(NAME)	append to old subpicture
ADPICT(INDEX)	same as ADDON (add by INDEX)
ADDOFF	end append subpicture mode
DIQPLA(NAME)	display already defined subpicture
DSPICT(INDEX)	display by INDEX number
ERSPIC(NAME)	erase already defined subpicture
ERPICT(INDEX)	same as ERSPIC (erase by INDEX)
EMPTYP(NAME)	empty contents of subpicture
SERCH(NAME,INDEX)	look up subpicture index

Subpictures give the user many additional capabilities not available on the Tektronix terminals. Subpictures allow the user to take advantage of the refresh aspect of the Imlac PDS-4 by using selective erase, redefining individual collections of vectors, and creating multiple copies of a single image with minimal data storage and minimal data transmission.

Subpictures may call other subpictures up to a level of six deep. Subpicture calls are valid only while in GRAPHICS mode. Once a program enters SUBPICTURE mode, all other valid GRAPHICS mode routines may be called.

5.1.1 Subpicture Identification

In GRAPHELP, each subpicture is identified by a unique index number or by a unique six-character Hollerith name. If a Hollerith name is used, then GRAPHELP attaches the name to a corresponding index number and keeps track of the "index number to name association" in an internal directory of its own. Although the user may have access to this directory and the corresponding index numbers, it is not advised to do so without a thorough understanding of the GRAPHELP subpicture maintenance system. Therefore, it is suggested that only one identification system be used at a time in a single program (either the index system or the name system) and that intermixing of the two systems be avoided. Note that an index number must be an integer between 1 and 128, inclusive.

5.1.2 Subpicture by Name

To create a subpicture by name, call OPENQ with an unused name. This call puts the program into SUBPICTURE mode. All display output from this point forward is channeled into this subpicture until CLOSEQ is called. Up to 128 different subpictures can be created. If OPENQ is called with a name that has already been used, then the old subpicture is replaced by the new incoming display information.

Subpicture names must always be six characters long, including blanks and special characters.

Example 12

The following are legal subpicture names:

```
1 CALL OPENQ(6HPICTUR)
2 CALL OPENQ('PICTUR')
3 CALL OPENQ(6HPIC1  )
4 CALL OPENQ('  PIC1')
5 CALL OPENQ(8HPICTURE1)     unique only to six characters

6 DIMENSION NAME(3)
  DATA NAME/6HPICTUR/
  CALL OPENQ(NAME)
```

Example 13

The following are illegal subpicture names:

```
1   CALL OPENQ('PIC1')       not enough characters
2   CALL OPENQ(3HPIC)        not enough characters
3   CALL OPENQ(PICTUR)       not in Hollerith format
```

To display a subpicture, the user calls DIQPLA with the name of the subpicture to display. To delete a subpicture, the user calls ERSPIC with the name of the subpicture. ERSPIC completely wipes out the subpicture. If the user wishes only to empty the contents of a subpicture, he calls OPENQ and immediately follows this with a CLOSEQ or else calls EMPTYP with the subpicture name. This call creates an empty subpicture. An *OPENQ call always redefines a new subpicture.* If no subpicture of that name exists, then a new one is defined. Subpictures are not displayed until DIQPLA is called. But DIQPLA cannot be called until the subpicture has been created.

To append to a subpicture, the user calls ADDON. This call forces the program to reenter SUBPICTURE mode and append the display information to the end of the already defined subpicture. ADDOFF ends ADDON mode.

Internally, each subpicture name is stored with an identifying index number. Each time ERSPIC is called, the subpicture's index number is freed and made available for the next new subpicture definition. The routine SERCH(NAME,INDEX) is used to look up a subpicture index number. If SERCH returns a -1 in INDEX, then the NAME passed to it does not exist in its directory. NAME takes the same format as described by OPENQ.

5.1.3 Subpicture by Index

Many times it is more useful to identify subpictures by a number rather than by a name. This identification usually requires less storage and no directory. For this reason, OPENQ, DIQPLA, ERSPIC, and ADDON have counterparts to identify subpictures by an index number. The routines are OPPICT, DSPICT, ERPICT, and ADPICT, respectively. They behave the same as their name-identifying counterparts. A subpicture may be opened by calling OPPICT(IPIC) and closed by calling CLOSEQ.

5.1.4 Attaching Subpictures

Subpictures identified by index number or Hollerith name have the additional capability of cross-hair attachment. Cross-hair attachment is an interactive facility provided to temporarily attach and move subpictures by using the GTS cross-hair cursor. It is enabled by calling ATPICT or ATACHP with the index number or Hollerith name of the appropriate subpicture. It may be detached by calling DEPICT or calling the attach operation with a different subpicture.

ATACHP(NAME)	attach subpicture to cross hair
ATPICT(INDEX)	same as ATACHP (attach by INDEX)
DEPICT	detach picture from cross hair

To make the attaching work, a subpicture is created by using OPPICT or OPENQ and closed with CLOSEQ. The display content of the subpicture must be composed only of relative vectors or text, to make the picture move. If absolute vectors are used, then that part of the subpicture is essentially fixed to the screen and does not move. The subpicture must always be made up of relative vectors, to make the subpicture position independent. Once the picture is created, it can be attached to the cross hair by using ATPICT or ATACHP. When the cross-hair cursor is displayed on the screen (i.e., CALL INQUIR), then a copy of the subpicture appears on the screen whose origin is attached to the center of the cross-hair cursor. As the cross-hair cursor moves, the image copy of the subpicture moves with it. This facility is handy to use for moving objects to different positions on the screen.

To use the subpicture control effectively, one should understand the Imlac PDS-4 intelligent terminal display file structure. Subpictures can be used to simulate the motion of an object by nesting subpictures and redefining a position coordinate. All subpicture redefinitions are double buffered in the Imlac PDS-4. More programming help of subpictures is offered by the examples in section 6 of this manual.

5.2 Data Tablet

GRAPHELP supports the graphic data tablet by incorporating three modes of operation. These modes are ALIGNMENT mode, ENQUIRY mode, and TRACE mode. All modes are programmable from FORTRAN and use the cross-hair cursor on the screen to reference tablet pen positions with a corresponding position on the screen. The current cross-hair position is always displayed in the upper right corner of the screen in decimal raster units. In general, when using the tablet, one should hold the pen as vertically as possible to get a fairly constant reading. The user should not place any obstruction between the pen and the sensors (e.g., his hand). To register a coordinate value, the user pushes the pen down so that the pen point moves in. If TRACE mode or ENQUIRY mode is in effect, there is usually a click of some sort upon registration. Sometimes there is a slight delay to get the pen to register. The user holds down the pen a little longer; by listening and thinking, he can sense a valid pen registration. Dead spots in the tablet occasionally occur near the edges and the corners. These spots should be avoided.

5.2.1 ALIGNMENT Mode

ALIGNMENT mode allows the user to align a picture on the tablet with positions on the screen. It is important to use this mode at the beginning of any tablet sessions, to insure that the picture is on straight with respect to the screen coordinate system.

Typing REP-H toggles this mode on and off. It should always be off before going to any other mode. In this mode, there is no communication between the tablet and the host computer. All functions are strictly local between the screen and the tablet.

When ALIGNMENT mode is on, a cross-hair cursor comes on the screen. It may be positioned by the arrow keys (normal GTS method) or with the tablet pen. The cursor always follows the position of the tablet pen. In this manner, one may align tablet positions with screen coordinates by moving the pen over the face of the tablet and observing the screen coordinates displayed in the upper right corner of the screen. Pressing the pen down has no effect in this mode.

5.2.2 ENQUIRY Mode

ENQUIRY mode allows the user to interrogate the current position of the cross hair under program control from a host computer. When initiated, the cross-hair cursor comes on the screen as in ALIGNMENT mode. The user may now put the cursor at any position desired and send the coordinates to the host computer using one of two methods:

a. He types a keyboard character (not carriage return).
b. He pushes down the pen point on the tablet pen.

Either action terminates ENQUIRY mode, and the cross hair leaves the screen. Pushing the pen down is synonymous with typing a T. The data transmitted appear as follows:

a. The character is typed on the keyboard in A1 format.
b. The cross hair coordinates in 2I4 format.

5.2.3 TRACE Mode

TRACE mode allows a user to send continuous coordinate data from the tablet to the host computer. When TRACE mode is initiated, x-y coordinate pairs of the current cross-hair position are transmitted whenever the tablet pen is pushed down. If the pen is not pushed down, no data are transmitted to the host.

If a local trace is enabled, the Imlac PDS-4 connects the points with the current line type enabled. The first depression of the pen tip is always drawn with an invisible move. The tablet has a built-in precision constant defaulted at one raster unit. This is the minimum distance that the tablet pen must move before the cross hair moves. All the trace parameters (enable, line type, and precision) are programmable from the host computer.

5.2.4 GRAPHELP Tablet FORTRAN Support

All tablet routines must be called while in GRAPHICS mode only.

SETTAB(IPREC,LINETP,IUNITS,ITYPE,IDISA)	set tablet parameters
INQUIR(X,Y,ICHR)	initiate ENQUIRY mode
ENQUIR(IX,IY,ICHR)	initiate ENQUIRY mode
TRACON(X,Y,N,N2,ICHR)	initiate TRACE mode
SCFTAB	scale tablet

SETTAB sets up the tablet parameter table. It is an optional call. If it is not called, default values are assumed. The defaults are SETTAB(1,1,1HR,4,1HE).

IPREC is the tablet precision where 0 < IPREC < 256.

LINETP is the line type to use in a local trace.

0	invisible
1	solid
2	dotted
3	dashed
4	dot-dashed

IUNITS is the unit type to return x-y values when in TRACE mode or ENQUIRY mode.

1HR	raster units
1HI	inches units
1HV	virtual units

ITYPE specifies the type of scaling in effect on the virtual coordinate system.

1	x logarithmic
2	y logarithmic
3	x-y logarithmic
4	x-y linear

IDISA is used to enable or disable the Imlac PDS-4 local trace.

1HE	enable
1HD	disable

INQUIR initiates ENQUIRY mode where X and Y are the position of the cross-hair cursor in the current specified IUNITS of SETTAB.

ICHR is the character typed on the keyboard. If the pen is used to terminate ENQUIRY, then ICHR is equivalent to 1HT.

ENQUIR works the same as INQUIR, except that the x-y coordinates are returned in raster units only in integer format.

SCFTAB is used to scale the virtual tablet space interactively from the tablet at run time. It requests an origin, a relative x displacement, and a relative y displacement to calculate the scaling space.

TRACON initiates Imlac PDS-4 tablet TRACE mode. It essentially reads x-y coordinate pairs from the tablet continuously whenever the pen is down. The x and y are dimensioned real buffers to return the coordinate data values. N is the size of the buffers, and N2 is the number of points actually read. If N2 is greater than N, then the last N2-N points have not been read. ICHR is the keyboard character typed to terminate TRACE mode.

6. EXAMPLES AND SUGGESTIONS

6.1 Examples

The following examples and suggestions are provided as aids to the reader to illustrate methods of using GRAPHELP efficiently for different classes of graphic problems. All examples should produce some kind of real output if implemented.

Example 14

Draw a 1-in. square on the screen.

```
CALL SETPDQ              set up graphics
CALL SCREEN              erase screen while in ALPHA mode
CALL ENTGRA              enter GRAPHICS mode
CALL ABSVEC(2.,2.,0)     position to 2.,2. beam off
CALL RELVEC(1.,0.,1)     begin drawing square
CALL RELVEC(0.,1.,1)
CALL RELVEC(-1.,0.,1)
CALL RELVEC(0.,-1.,1)
CALL EXITGR              exit GRAPHICS mode
STOP
END
```

To draw the same square with the sides one half the length of the 1 in., insert a CALL FACTOR(.5,.5) statement right after the call to ABSVEC. This call reduces the overall output to the screen in both the

x and y directions. To draw the same square distorted as a rectangle with x twice y, change factor parameters to CALL FACTOR(1.,.5) or any ratio of x to y that is 2:1. To draw the square with dotted lines, change the last parameter in RELVEC from 1 to 2 or CALL RELVEC (X,Y,2).

Example 15

Draw three squares on the screen.

There are two good ways to write the subroutines that draw them. One way is to use subpictures, but this is compatible only with the Imlac PDS-4. The other is to use multiple calls to a square subroutine for redrawing a square three times. The first method is more efficient because it transfers fewer data. Its disadvantage is that it cannot be used on the Tektronix. The two subroutines in this example demonstrate the two different methods.

```
      SUBROUTINE METH1(XPO,YPO)
C
C  DRAW A SQUARE USING SUBPICTURES ASSUMING SUBPIC DEFINED ALREADY
C  POSITION SQUARE AT XPO,YPO WITH SUBPICTURE NAME   'SQUARE'
      CALL ABSVEC(XPO,YPO,0)     position square
      CALL DIQPLA('SQUARE')      display subpicture
      RETURN
      END

      SUBROUTINE METH2(XPO,YPO)
C
C  DRAW A SQUARE AT POSITION XPO,YPO
C
      CALL ABSVEC(XPO,YPO,0)     position square
      CALL RELVEC(1.,0.,1)       begin drawing square
      CALL RELVEC(0.,1.,1)
      CALL RELVEC(-1.,0.,1)
      CALL RELVEC(0.,-1.,1)
      RETURN
      END
```

In METH1, only one data word is transmitted for each call to the routine. In METH2, the four sides of the square are transmitted each time. This difference shows that as the object's complexity and frequency of transmission rises, the efficiency of the data transfer drops, and METH1 becomes more and more ideal for fast graphics. This fact is particularly true at lower communication speeds.

The calling program for METH1 might look something like this:

```
        CALL SETPDQ                 set up graphics
        CALL SCREEN                 erase screen while in ALPHA mode
        CALL ENTGRA                 enter GRAPHICS mode
C
C  DEFINE THE SUBPICTURE CALLED SQUARE
C
        CALL OPENQ('SQUARE')        open subpicture definition
        CALL RELVEC(1.,0.,1)        begin drawing square
        CALL RELVEC(0.,1.,1)
        CALL RELVEC(-1.,0.,1)
        CALL RELVEC(0.,-1.,1)

        CALL CLOSEQ                 close subpicture definition
C
C  DISPLAY THE SQUARE IN 3 DIFFERENT PLACES
C
        DO 10 I=1,3
10      CALL METH1(FLOAT(I),FLOAT(I))
C
        CALL EXITGR                 exit GRAPHICS mode
        STOP
        END
```

The calling program for METH2 looks the same with the omission of the lines bounded by OPENQ and CLOSEQ inclusive. If ERASEQ is called to erase the screen at the end of the program, the square can be restored by a single call to DIQPLA if METH1 is used. However, METH2 requires that the entire square be redrawn.

Example 16

Move a square from one corner of the screen to another (pseudomotion simulation).

```
        CALL SETPDQ                 set up graphics
        CALL SCREEN                 erase screen while in ALPHA mode
        CALL ENTGRA                 enter GRAPHICS mode
C
C  DEFINE THE SUBPICTURE CALLED SQUARE
C
        CALL OPENQ('SQUARE')        open subpicture definition
        CALL RELVEC(1.,0.,1)        begin drawing square
        CALL RELVEC(0.,1.,1)
        CALL RELVEC(-1.,0.,1)
        CALL RELVEC(0.,-1.,1)
        CALL CLOSEQ                 close subpicture definition
```

```
C  DEFINE A 'POSITION-THE-SQUARE' SUBPICTURE AS EMPTY

C
        CALL OPENQ('POSSQU')
        CALL CLOSEQ
        CALL DIQPLA('POSSQU')
C
C  REDEFINE POSSQU WITH SQUARE IN A NEW POSITION IN A LOOP
C
        CALL FACTOR(.2,.2)          move square at 0.2-in. increments
        DO 10 I=1,50
        X=FLOAT(I-1)
        CALL OPENQ('POSSQU')        redefine square position subpicture
        CALL ABSVEC(X,X,0)          put square at position x,x
        CALL DIQPLA ('SQUARE')      display square
        CALL CLOSEQ                 close redefinition
10      CONTINUE
        CALL EXITGR
        STOP
        END
```

This program essentially redefines a subpicture called POSSQU each time through a loop. Each new definition simply redraws the SQUARE at a new position in 0.2-in. increments. The effect is a square sliding diagonally up the screen. The speed is a function of the transmission rate and the increment size.

Example 17

Use the PROMPT to suspend a program at run time. Repeat example 16, but suspend motion when a 1 is in the Imlac PDS-4 switch register.

This example is an illustration of suspending a program during run time without halting program execution. This suspension allows the user to examine or change picture parameters at run time without stopping the program or exiting GRAPHICS mode. To suspend the program, the user inserts the following lines before the continue statement of example 16.

```
5       CALL PROMPT(KSWIT)          read switch register
        IF(KSWIT.EQ.1) GO TO 5      if it is "1," read it again
```

This insertion loops the program indefinitely while a "1" is in the switch register. Other actions can be initiated besides looping. This routine with INQUIR and IWAITQ gives the programmer a wide variety of possible run-time interactions. Up to eight different switch settings can be read from the switch register (three bits).

Example 18

Read the cross hair and draw a line to it on the following input:

```
    I   invisible
    S   solid
    Q   quit program
```

```
      DATA IHI,IHS,IHQ/1HI,1HS,1HQ/
      CALL SETPDQ              initialize
      CALL SCREEN              erase screen
      CALL FACTOR(0.,0.)       set factor to screen resolution
      CALL ENTGRA              enter GRAPHICS mode
      CALL WILDCR(2)           make ALPHA list invisible
10    CALL INQUIR(X,Y,I)       get cross-hair coordinate
C
C  CHECK INPUT CHARACTER
C
      IF(I.EQ.IHQ)GO TO 20
      IF(I.NE.IHI.OR.I.NE.IHS)GO TO 10
C   SET  LINE TYPE
      LINE=0
      IF(I.EQ.IHS)LINE=1
      CALL ABSVEC(X,Y,LINE)      draw vector to x,y
      GO TO 10                   get next vector
20    CALL EXITGR
      STOP
      END
```

In example 18, FACTOR sets the 1-in.-length equivalent to a single screen raster unit. The call to WILDCR makes the ALPHA display disappear on the Imlac PDS-4 without affecting the GRAPHICS display. It does nothing on the Tektronix terminals. INQUIR is used to display the cursor and get interactive coordinates from the user. The character typed on the keyboard determines the nature of the following action.

Example 19

Using general graph GENGRF, generate a data plot.

```
      DIMENSION X(50),Y(50)
C
C  GENERATE DATA PLOT Y=X
C
      DO 10 I=1,50
      X(I)=FLOAT(I-1)
        Y(I)=X(I)
10    CONTINUE
```

```
CALL GENGRF(X,Y,50)    draw linear graph of y=x (50 points)
STOP
END
```

This plots a linear graph of x=y using GENGRF to do all graphics. No call to SETPDQ, EXITGR, FACTOR, or any other routine is necessary. At run time, GENGRF asks for titles and other pieces of information.

Figure 1 shows a typical scenario using GENGRF with example 19. The underlined statements were typed in by the user. GENGRF uses default values if no explicit value is returned (carriage return). Also, values outside the specified range force the default values to override the typed in input. Figure 2 shows the graph that was created by this scenario. Figure 3 shows the result of typing p when the graph was finished. The cross hair came on the screen, and the title was repositioned by positioning the cross hair and striking a random character on the keyboard.

```
   HOWDY. AHM GENERAL GRAPH!
JUST FILL IN THE BLANKS
AND DONT FERGIT DECIMAL POINTS IN AXIS LENGTHS
ILL WAIT FER YOU WHEN AHM DONE DRAWIN
AT THAT TIME TYPE ONE OF THE FOLLOWING

R-REDO
P-POSITION TITLE
M-MOVE GRAPH
ANYTHING ELSE-QUIT

INPUT  2 <XAXIS LENGTH<10 (I DEFAULT 5. INCHES)
5.
INPUT  2 <YAXIS LENGTH<10 (I DEFAULT 5. INCHES)
<CR>
IF YOU WANT A GRID TYPE   G
G
INPUT XAXIS TITLE UP TO 20 CHARACTERS
XAXIS
INPUT YAXIS TITLE UP TO 20 CHARACTERS
YAXIS
INPUT TITLE OF GRAPH UP TO 30 CHARACTERS
TITLE
```

Figure 1. Use of GENGRF.

Figure 2. Typical GENGRF graph.

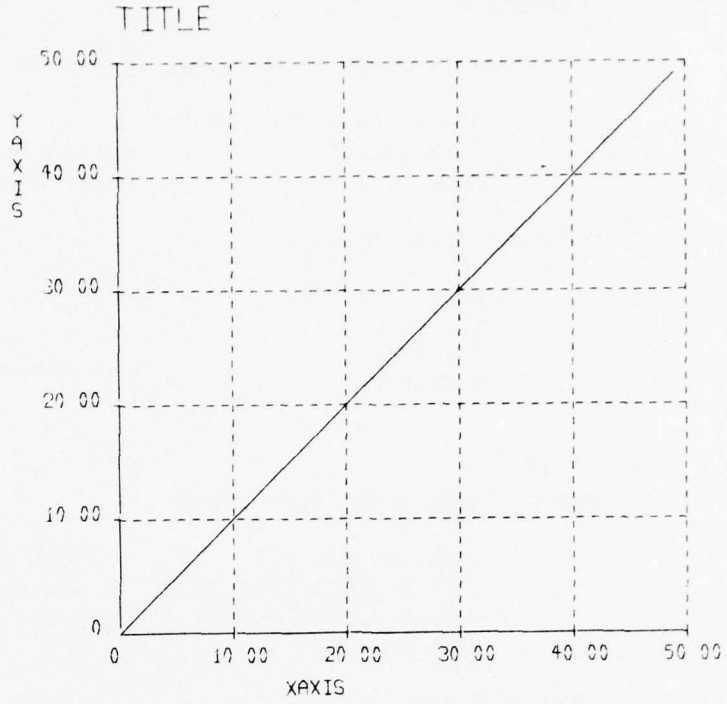
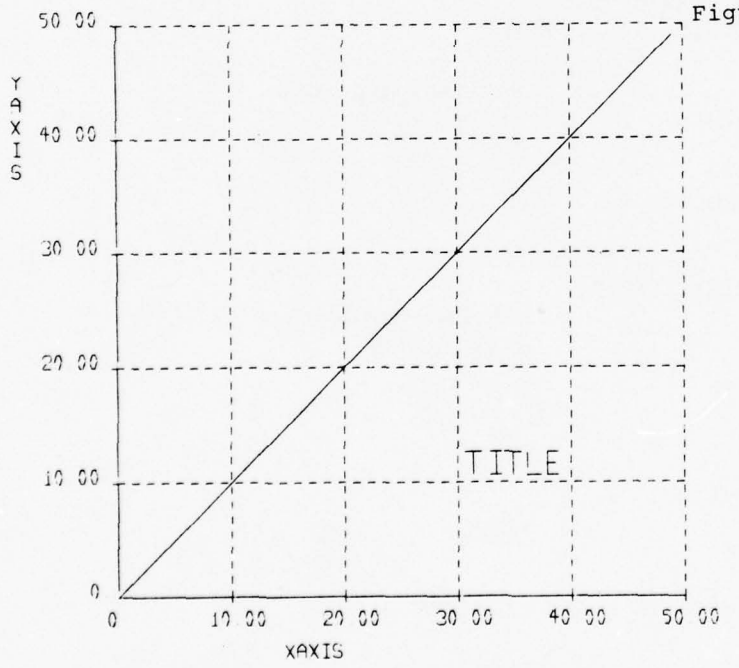


Figure 3. Result of GENGRF graph.



Example 20

Draw a graph with a plot symbol at each point.

```
      DIMENSION X(50),Y(50)
      CALL SETPDQ                initialize
      CALL SCREEN                erase screen
C
C  GENERATE DATA PLOT Y=X
C
      DO 10 I=1,50
      X(I)=FLOAT(I)
      Y(I)=X(I)
10    CONTINUE


C
C  SET UP IMAGE SPACE PARAMETERS
C
      CALL INIT(2.,2.)           origin at 2,2
      CALL SCALE(X,50,5.,1,0)    x scale on 5-in. axis
      CALL SCALE(Y,50,5.,1,1)    y scale on 5-in. axis
      CALL ENTGRA
C
C  DRAW AXIS AND A GRID
C
      CALL XAXIS('XAXIS',5,5.)
      CALL YAXIS('YAXIS',5,5.)
      CALL GRID(5.,5.,1.,1.,4,0)
C
      LINE=3                     set line type off
      DO 20 I=1,50
C

C  PLOT DATA WITH SYMBOL


  CALL DRAW(X(I),Y(I),LINE,4)
  LINE=2                         set line type solid
  CALL PLSYMB                    draw symbol
20 CONTINUE
  CALL EOFXMT
  STOP
  END

C
  SUBROUTINE PLSYMB
C
C  SAMPLE PLOTTING SYMBOL IS A BOX OF SIDES .04
```

```
CALL RELVEC(-.02,-.02,0)
CALL RELVEC(.04,0.,1)          begin drawing square
CALL RELVEC(0.,.04,1)
CALL RELVEC(-.04,0.,1)
CALL RELVEC(0.,-1)
CALL RELVEC(.02,.02,0)
RETURN
END
```

In example 20 the graph y = x is plotted with a box drawn from the center placed at each data point. PLSYMB can be written to draw any picture desired at any size or frequency. DRAW is used to plot the data point by point. PLSYMB was drawn from the center of the symbol and ends there. In addition, PLSYMB is drawn with relative vectors to make the symbol position independent. Also, PLSYMB is drawn with RELVEC, rather than RDRAW so that the scaling of PLSYMB is not affected by the virtual space parameters. Figure 4 shows the output of this program.

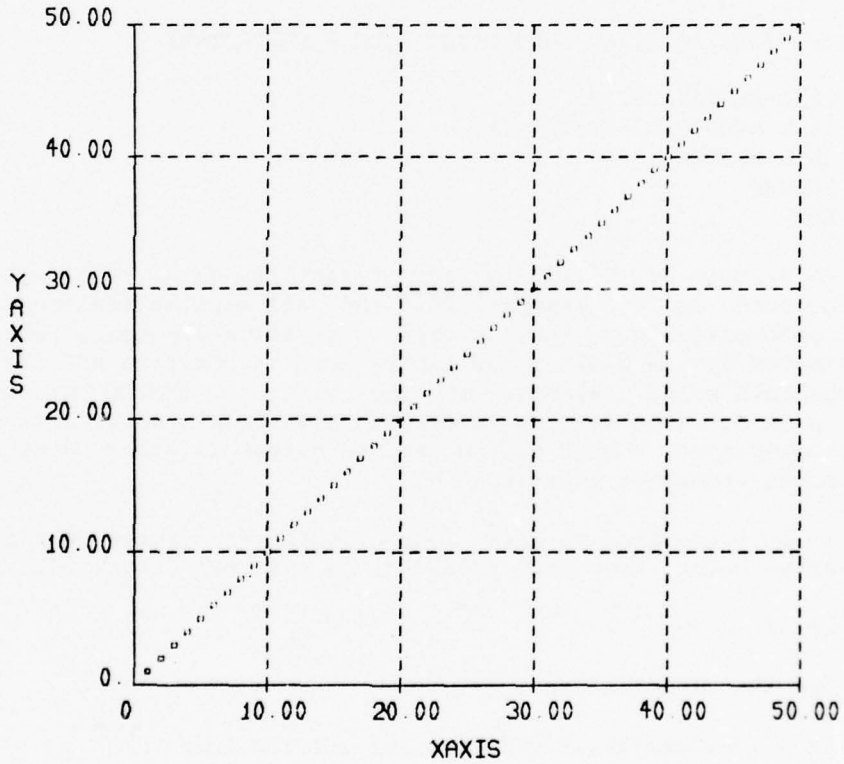


Figure 4. Output of PLSYMB.

Example 21

Draw a y-log, x-linear graph with a title, assuming that the data are in arrays x,y of size N.

```
      SUBROUTINE SEMLOG(X,Y,N)
C
      DIMENSION X(1),Y(1)
      CALL SETPDQ
C
      CALL INIT(2.,2.)              origin at 2,2
      CALL SCALE(X,N,5.,1,0)        x scale on 5-in. axis
      CALL LOGQCA(Y,N,5.,1,1)       y scale on 5-in. axis
      CALL ENTGRA C
C  DRAW AXIS AND DATA PLOT
C
      CALL XAXIS('XAXIS',5,5.)
      CALL YLOGAX('YAXIS',5,5.)
      CALL DATLOG(X,Y,N,1,4,3)

C
C  POSITION TITLE AT 2.,6.  AND PRINT SIZE 6 CHARACTERS
C
      CALL PLOT(2.,6.,3)
      CALL SYMBOQ(5,5HTITLE,6)
      CALL EXITGR
      RETURN
      END
```

In example 21, *DATLOG is used to plot the data.* Alternatively, DRAW can be used as in example 20, and all semilog scaling is then performed automatically. PLOT is used to position the text, because it is not affected by the scaling and can be used to position off the graph at some absolute point relative to the origin. SYMBOQ puts out the title at size 6. (The size is ignored on a Tektronix 4012). If values need to be displayed, NUMBRQ can be used to output floating point binary with no format statement necessary.

If it is desired to have four significant digits to the right of the decimal point, then CALL AXPREC(4) is inserted before CALL XAXIS.

Example 22

Use the clipping facility.

```
      DATA IHS,IHI,IH1,IH2,IH3/1HS,1HI,1H1,1H2,1H3/
      CALL SETPDQ                   initialize GRAPHELP
      CALL SCREEN                   erase screen
```

```
C  SET VIRTUAL SPACE UP
         CALL INIT(1.,1.)
         CALL QCALE(0.,120.,6.,SF,VLO,0)     set up scaling
         CALL QCALE(0.,120.,6.,SF,VLO,1)     and a default window
C  NOTE THAT WINDOW IS FROM 0,0 to 120,120
C
         CALL ENTGRA
         CALL XAXIS('XAXISCLIP',9,6.)        draw axes 6 in. long
         CALL YAXIS('YAXISCLIP',9,6.)
C  INITIALIZE THE BEAM TO ORIGIN
C
         CALL DRAW(8.,0.,3,4)
C
C  READ CROSSHAIR POSITION CONVERT TO VIRTUAL COORDINATES
10       CALL INQUIR(X,Y,ICH)
         CALL RSVIRT(X,Y,X1,Y1,4)
C  DRAW THE CLIPPED LINE TO THE CURSOR POSITION
         IF(ICH.EQ.IHS)CALL DRAWC(X1,Y1,1,4)      solid line
         IF(ICH.EQ.IH1)CALL DRAWC(X1,Y1,4,4)      dotted
         IF(ICH.EQ.IH2)CALL DRAWC(X1,Y1,5,4)      dashed
         IF(ICH.EQ.IH3)CALL DRAWC(X1,Y1,6,4)      dot-dashed
         IF(ICH.EQ.IHI)CALL DRAWC(X1,Y1,3,4)      invisible
         IF(ICH.EQ.IHQ)GO TO 50                   quit
         GO TO 10
50       CALL EXITGR
         STOP
         END
```

Example 22 demonstrates a simple use of the GRAPHELP clipping facility. The virtual coordinate system is established upon an area 6 × 6 in., where the limits run from 0.0 to 120.0 units. In addition to scaling these units over an area 6 × 6 in., the two calls to QCALE set up default window boundaries at (0.0,0.0) and (120.0,120.0). The axes are drawn at the lower boundaries and are not clipped when drawn. Each call to INQUIR returns an x-y coordinate value in raster units. The call to RSVIRT converts this coordinate value to the scaled virtual coordinate system (0 to 120) and stores the values in X1 and Y1. The remaining section draws a line to the cross hair with a line type that is determined by whatever key the user typed on the keyboard. A "Q" terminates the program. If the cross-hair cursor is positioned outside of the 6 × 6 in. area, the call to RSVIRT generates values that are outside of the scaled coordinate system (i.e., greater than 120.0 or less than 0.0). The resulting line is clipped at the nearest boundary. The next call to INQUIR generates a line beginning at the imaginary position where the line would have been if it had not been previously clipped.

If DRAW is used in example 22 rather than DRAWC, then the lines are not clipped at the 0 to 120 boundary and are drawn to wherever the cursor is placed.

Sometimes the default window boundary is not desirable. To change the default clipping window in example 22 to clip all x values outside of 50.0 to 98.5 and all y values outside of 10.2 to 75.7, the call SETWIN(50.0,98.5,10.2,75.7) can be inserted after the two calls to QCALE. If INQUIR is used many times in many different places in a program and the virtual coordinate system is being used, it is often easier to force INQUIR to return the x-y values in scaled virtual units rather than in raster units. In this way, the call to RSVIRT is unnecessary. Example 22 may be modified to illustrate this point by the following steps.

a. Add after the call to SCREEN, CALL SETTAB(1,1,1HV,4,1HE).

b. Change CALL INQUIR(X,Y,ICH) to CALL INQUIR(X1,Y1,ICH).

c. Eliminate the call to RSVIRT.

Example 23

Digitize data using the tablet.

```
      SUBROUTINE GATHER(X,Y,N,N2,ICHR)
      DATA IRETRY/1HR/
C
C  GATHER DATA FROM THE TABLET INTO X AND Y
C
C  X,Y - ARRAYS TO RETURN DATA
C   N  - LENGTH OF ARRAYS
C  N2  - RETURNED VALUE OF ACTUAL NUMBER OF POINTS GATHERED
C ICHR - CHARACTER TYPED AT KEY BOARD TO END GATHER
C
      DIMENSION X(1),Y(1)
      CALL ENTGRA              enter GRAPHICS
C
C  SETUP THE TABLET
      CALL SCFTAB              scale from tablet
   CALL SETTAB(1,3,1HV,4,1HE)  enable trace, set line type as dashed
C
C  TRACE ON THE TABLET
10    CALL ERASEQ                  erase screen
      CALL TRACON(X,Y,N,N2,ICHR)   gather data
      IF(ICHR.EQ.IRETRY)go to 10   if R is typed, try again
      CALL EXITGR
      RETURN
      END
```

Subroutine GATHER traces and gathers data on the sonic data tablet. If the character "R" is typed to end the trace, GATHER erases

the screen and tries again. GATHER assumes that SETPDQ was invoked previously by the calling program. Also, if on return N2 is greater than N, then an attempt was made to gather more points than the capacity of the dimensioned array. The error is not fatal, because TRACON stops collecting data after N points, and the value N2-N indicates how many points were lost.

A typical calling program for GATHER opens a disk file, calls GATHER, analyzes and refines the collected data, writes the data onto the disk file, calls GATHER again, and continues the cycle until the user types a "Q" on the keyboard to "quit" the process. The "Q" is returned in ICHR. Each call to GATHER traces the data locally on the CRT screen with a dashed line. This trace can be disabled by calling SETTAB with a 1HD instead of a 1HE in the last parameter.

6.2 Suggestions

Because it is not possible to cover everything in one manual, this section is provided as a library of aids to make programming graphics with GRAPHELP a little bit easier. As the user reads this manual, he does not have to understand everything, but he should at least get a general idea of how interactive computer graphics are implemented using GRAPHELP.

For most applications, the routines in the two-dimensional plotting package (sect. 4) are sufficient when used with a few of the control functions described in section 3. The two-dimensional functions let a user establish his own coordinate system, and the control functions let him control some of the human interaction. In fact, it is rarely necessary to have to use routines ABSVEC and RELVEC. If a user establishes his own coordinate system using the scaling routines, DRAW and RDRAW do all of his scaling and translations. To insure that his picture does not draw off of the usable area on the screen, he should use RDRAWC and DRAWC to perform the desired clipping. This facility is especially important when he is using the Tektronix storage tube terminals as his output device. These terminals do not contain the hardware clipping facility that is otherwise available in the Imlac PDS-4.

Drawing vectors with DRAW and RDRAW is also more convenient when lines must be drawn in a logarithmic or semilogarithmic scaled area. RELVEC and ABSVEC become very clumsy to use when a user has to rescale each vector from his units to the screen units. This facility becomes particularly useful for the application of data plotting in two dimensions.

Screen control is very important in graphics when using a CRT as the output device. For this reason, several methods for erasing the

screen are provided in GRAPHELP. Screen erasing techniques can be classified into two different categories: unconditional and conditional.

Unconditional screen erasing is provided by SCREEN, ERASEA, ERASEQ, and ERSALL. SCREEN provides the user program with a means to erase the screen when no graphics are being performed. The most-often-used application for SCREEN is at the beginning of a program to clear the screen before GRAPHICS mode is entered. This clearing insures that the program is starting with a clear working area on the screen. On the Tektronix terminals, the remaining three ERASE routines are synonymous. But on the Imlac PDS-4, each ERASE routine has a slightly different effect on the CRT screen.

Many interactive applications require messages to be displayed on the screen to instruct the user to take a particular action. If this text is displayed in ALPHA mode, it can be erased without disturbing the GRAPHICS file. A typical program exits GRAPHICS, uses a FORTRAN WRITE to display the message, waits for a user input, enters GRAPHICS mode, erases the message only, and appends to the graphics display with more lines and symbols.

The preceding sequence can be viewed as a conditional type of screen ERASE where the condition is a function of the user input. Because this type of programming sequence occurs so frequently, IWAITQ is provided in GRAPHELP as an interactive conditional ERASE. It has seven different op codes to perform slightly different functions between the terminals. A common mistake is to use IWAITQ while still in GRAPHICS mode. The user should always call IWAITQ while in ALPHA mode, or the results are completely unpredictable. The most common use of IWAITQ is on the Tektronix terminal. A typical program draws a picture, exits GRAPHICS mode, and calls IWAITQ. This program allows the user to view the program as long as required and then to stop or take some other action.

Although it is desirable to write a program that one can display on any terminal, sometimes it is necessary to write a program that specifically uses only the refresh terminal (i.e., graphic editing). Although it is true that any program written for the storage tube runs correctly on the refresh tube, the reverse is not always possible. None of the subpicture routines have an effect on the storage tube. Besides the subpicture capability, some other features include selective blink, variable intensity, and 90-day rotations. One way of distinguishing objects on the screen is to draw them with different intensities or with different line types. A good method of making one object on the screen stand out from all the other objects is to make it blink on and off periodically. Finally, since 90-deg rotations are possible in hardware on the Imlac PDS-4, drawing text vertically is possible by calling ROTA90 and then calling SYMBOQ with the text.

When using subpictures, always remember that a subpicture cannot be moved locally in the Imlac PDS-4 unless it is drawn with relative vectors. An absolute vector glues that part of the picture to the screen. If a user wants to move that subpicture, then he must retransmit the entire subpicture. This retransmission is unnecessary if relative vectors are used (e.g., RELVEC, RDRAW, RDRAWC, RYAXIS, and RXAXIS).

Subpictures may be used to give the appearance of movement by taking advantage of the double buffering effect on redefinition. When using subpictures identified by name, the user should be warned that ERSPIC erases only the subpicture and does not remove the calls to the picture. In addition, the index number for that subpicture is made available for the creation of the next subpicture invoked by OPENQ. Thus, if the user does not eliminate the old calls, the newly created subpicture using an old index number appears on the screen without any calls to DIQPLA.

When using relative vectors on the refresh tube, roundoff errors begin to accumulate. RELVEC and ABSVEC internally account for these roundoff errors. In addition, all calls to subpictures identified by name keep track of these roundoff errors. Any calls to ABSVEC resets the error to zero. Many times, these small roundoff errors are insignificant and such tabulations are not needed. Thus, to save storage, a reference to subpictures by index number rather than name does not keep track of the roundoff errors. This saving becomes important when program size is critical. Identifying subpictures by index number significantly reduces the amount of storage required because no directories are kept for the names or the roundoff errors. Of course, there is a potential error if the roundoff errors are critical.

Actual use of GRAPHELP requires that the routines be linked to the user's program. Appendix F contains directions for doing that on two different systems at the Harry Diamond Laboratories.

APPENDIX A.--CHANGES FROM VERSION 1 OF GRAPHELP

The following changes have been made to GRAPHELP since the release of version 1.

1. RESCA, DRAWC, RDRAWC, SETWIN, RSVIRT, INVIRT were added to the two-dimensional plotting package. These enhancements include rescaling over multiple arrays, clipping and setting windows (up to 7). New conversion routines allow data to be transformed from a raster- to an inches- to the user-scaled coordinate system.

2. With these additions came SETTAB to allow the user to specify what coordinate system INQUIR will return values in and to set tablet functions for the Imlac PDS-4 data tablet.

3. Support for the data tablet includes SETTAB, SCFTAB, and TRACON.

4. Subpicture identification by index number was added in the form of the routines OPPICT, ADPICT, DSPICT, and ERPICT. In addition, the routines ATPICT, ATACHP, EMPTYP, and DEPICT were added for cross-hair attachment and detachment.

5. Hard copy capability has been added to support the Versatec in conjunction with the Imlac PDS-4.

APPENDIX B.--LOADING GRAPHELP LIBRARIES

The following sequences assume that the user program is called PROG.

B-1. PRIME LOAD SEQUENCE

The following sequence should be used to load GRAPHELP on the PRIME operating system:

```
LOAD          enters loader
LO B PROG     load your program
LI IMGRAF     load for Imlac graphics only
   or
LI TKGRAF     load for Tektronix graphics only
   or
LI ALGRAF     load for graphics to either device
LI            load system library
SA *PROG      save program
QU            exit loader
```

B-2. PDP 11/45 RSX-11D TASK-BUILDING SEQUENCE

```
TKB           enter task builder
  Load your program and the graphics library assuming that
  GRAPHELP is in UFD [1.10].


PROG,TI:/SH=PROG,[1,10]IGRAPHELP/LB    link Imlac-only routines
      or
PROG,TI:/SH=PROG,[1,10]TGRAPHELP/LB    link Tektronix-only routines
      or
PROG,TI:/SH=PROG,[1,10]GRAPHELP/LB     link both device routines
/
ASG=TI:1              graphics input/output logical unit is always 1
LIBR=SYSRES:RO        declare system resident library
/
```

APPENDIX C.--PRIMITIVES OF GRAPHELP

The following is a catalogue of available graphic primitives contained in GRAPHELP. Appendix E contains a list of the two-dimensional plotting aids that use these primitives. Definitions of the calling parameters follow each list of routines.

C-1. CONTROL FUNCTIONS

SETPDQ	initialization of GRAPHELP
ENTGRA	enter *GRAPHICS* mode
EXITGR	exit GRAPHICS mode
ERASEQ	erase screen
ERASEA	erase ALPHA display
ERSALL	erase screen, clear all subpictures
SCREEN	erase screen while in ALPHA mode
EOFXMT	exit GRAPHICS, no more output till terminal key hit
FACTOR(XFACT,YFACT)	set overall output size

C-2. DRAWING ROUTINES

ABSVEC(XAB,YAB,LINETP)	draw absolutely to XAB,YAB
RELVEC(X,Y,LINETP)	draw relative to current location
SYMBOQ(NCHAR,ITEXT,NSIZE)	draw text in GRAPHICS

The following five are invoked ONLY on the Imlac PDS-4:

CIRQLE(*RADIUS,LINETP*)	*draw circle* (Imlac hardware only)
ARC(XCR,YCR,ANGLE,LINETP)	draw arc (Imlac hardware only)
ROTA90(N90)	rotate in 90-deg increments
INTENS(IBRITE)	change screen level intensity
BLINKQ	toggle blink on and off

C-3. SUBPICTURE CONTROL (IMLAC-PDS 4 ONLY)

OPENQ(NAME)	define new subpicture (redefine old one)
OPPICT(INDEX)	same as OPENQ (open by INDEX)
CLOSEQ	end define subpicture mode
ADDON(NAME)	append to old subpicture
ADPICT(INDEX)	same as ADDON (add by INDEX)
ADDOFF	end append subpicture mode
DIQPLA(NAME)	display already defined subpicture
DSPICT(INDEX)	display by INDEX number
ERSPIC(NAME)	erase already defined subpicture
ERPICT(INDEX)	same as ERSPIC (erase by INDEX)
EMPTYP(NAME)	empty contents of subpicture
SERCH(NAME,INDEX)	look up subpicture index
ATACHP(NAME)	attach subpicture to cross hair by name
ATPICT(INDEX)	attach subpicture to cross hair by INDEX
DEPICT(INDEX)	detach subpicture from cross hair

APPENDIX C

C-4. MISCELLANEOUS FUNCTIONS

WILDCR(NWILD)	Imlac, GTS/ASC wild-card mode
PROMPT(ISWIT)	interrogate Imlac data register
INQUIR(XRAS,YRAS,ICHAR)	turn on and get cross-hair position
ENQUIR(IXRAS,IYRAS,ICHAR)	same as INQUIR except integer values
DELAQ(NSEC)	CPU delay in seconds
DELAY(NULLS)	output nulls for delay
RSINCH(XRAS,YRAS,XAB,YAB)	convert rasters to inches
INVIRT(XAB,YAB,XS,YS,ITYPE)	convert inches to virtual
RSVIRT(XRAS,YRAS,XS,YS,ITYPE)	convert raster to virtual
FUNCTION IWAITQ(IOPCO)	prompt, wait, and erase
HDCOPY(ISWIT)	make hard copy of GRAPHICS file
SETPAG(XAB)	set CalComp end of page
SETTAB(IPREC,LINETP,IUNITS, ITYPE,IDISA)	set INQUIR (or TRACE) parameters
TRACON(XARA,YARA,NPTS,NRET, ICHAR)	initiate TRACE mode (Imlac only)
SCFTAB	scale tablet (Imlac only)

C-5. PARAMETER DEFINITIONS

XAB,YAB	absolute coordinate address in inches
X, Y	relative displacement in inches (NOTE: XAB, YAB, X, Y are relative to FACTOR)
LINETP	line type to draw where 0 beam off 1 beam on solid 2 beam on dotted 3 beam on dashed 4 beam on dot-dashed
NCHAR	number of characters to draw
ITEXT	text buffer (ASCII packed two characters/word)
NSIZE	size of character (1 to 7) where 1 is smallest and 7 is largest and 0 is terminal default size
N90	number of 90-deg rotations N90 > 0 counterclockwise N90 = 0 reset to 0 deg N90 < 0 clockwise
IBRITE	intensity scale (1 to 16) where 16 is brightest
RADIUS	circle radius in inches
XCR,YCR	center of arc relative to current position (inches)
ANGLE	angle magnitude and direction in radians ANGLE > 0 counterclockwise ANGLE = 0 full circle ANGLE < 0 clockwise
NAME	six-character Hollerith code identifying subpicture
INDEX	integer identifier associated with subpicture name

XFACT,YFACT — output size of total picture where
1.0 is full scale (1.0 = 1 in.)
2.0 is double scale (1.0 = 2 in.)
0.0 < FACT < 1.0 proportionally scale down picture
If FACT ≤ 0.0, then go to raster unit mode where
1.0 = 1 raster unit.

NWILD — wild-card identifier (see sect. 3.4 in body of report for list)

ISWIT — integer value of switch register (1 to 7)

XRAS,YRAS — absolute coordinate address in raster units

IXRAS,IYRAS — absolute coordinate in integer raster units

XS, YS — absolute coordinate in user-scaled (virtual) units

ICHAR — returned character typed on keyboard

NSEC — number of seconds to wait (approximation)

NULLS — number of nulls to output

IWAITQ — character typed from keyboard

IOPCO — op code for waiting type where

IOPCO	TEKTRONIX ACTION	IMLAC ACTION
0	wait and erase	nothing
1	prompt, wait, and erase	nothing
2	wait and erase	wait and erase
3	prompt, wait, and erase	prompt, wait, and erase
4	wait, no erase	wait, no erase
5	prompt, wait, no erase	prompt, wait, no erase
6	wait and erase	wait and no erase

IPREC — tablet precision where 0 < IPREC < 256

IUNITS — unit type to return X-Y values when in TRACE mode or ENQUIRY mode where
1HR raster units
1HI inches units
1HV virtual units

ITYPE — specifies type of scaling in effect on virtual coordinate system where
1 x log axis
2 y log axis
3 x-y log axes
4 x-y linear axes

IDISA — used to enable or disable Imlac PDS-4 local trace where
1HE enable
1HD disable

XARA,YARA — buffers to store X-Y data collected during trace

NPTS — maximum number of points for TRACON to collect

NRET — value returned by TRACON of number of points collected

APPENDIX D.--GRAPHELP CATEGORIZATION LIST

The following list divides the routines into three categories.

D-1. ROUTINES THAT MUST BE CALLED IN GRAPHICS MODE ONLY

ABSVEC(XAB,YAB,LINETP)	draw absolutely to XAB,YAB
ADDOFF	end append subpicture mode
ADDON(NAME)	append to old subpicture
ADPICT(INDEX)	same as ADDON (add by INDEX)
ATACHP(NAME)	attach subpicture to cross hair by name
ATPICT(INDEX)	attach subpicture to cross hair by INDEX
ARC(XCR,YCR,ANGLE,LINETP)	draw arc (Imlac hardware only)
BLINKQ	toggle blink on and off
CIRQLE(RADIUS,LINETP)	draw circle (Imlac hardware only)
CLOSEQ	end define subpicture mode
DATAQ(XARA,YARA,NPTS,INC,IPEN)	plot linear data
DATLOG(XARA,YARA,NPTS,INC, IPEN,ITYPE)	plot data
DALAQ(NSEC)	PRIME delay in seconds
DELAY(NULLS)	output nulls for delay
DEPICT(INDEX)	detach subpicture from cross hair
DIQPLA(NAME)	display already defined subpicture
DRAW(XS,YS,IPEN,ITYPE)	draw absolute in current scaled units
DRAWC(XS,YS,IPEN,ITYPE)	same as DRAW with clipping
DSPICT(INDEX)	display by INDEX number
EMPTYP(NAME)	empty the contents of the subpicture
ERASEA	erase ALPHA display
ERASEQ	erase screen
ERPICT(INDEX)	same as ERSPIC (erase by INDEX)
ERSALL	erase screen, clear all subpictures
ERSPIC(NAME)	erase already defined subpicture
EOFXMT	exit GRAPHICS, ignore all further output till key in
EXITGR	exit GRAPHICS mode
GRID(XGRD,YGRD,XD,YD,IPEN,IREL)	draw linear grid
HDCOPY(ISWIT)	make hard copy of GRAPHICS file
INQUIR(XRAS,YRAS,ICHAR)	turn on and get cross-hair position
INTENS(IBRITE)	change screen level intensity
NUMBRQ(VAL,IPREC,ISIZE)	draw floating point number
OPENQ(NAME)	define new subpicture (redefine old one)
OPPICT(INDEX)	same as OPENQ (open by INDEX)
PLOT(XIN,YIN,IPEN)	plot absolutely to inches relative to origin

PROMPT(ISWIT)	interrogate Imlac data register
RDRAW(XS,YS,IPEN,ITYPE)	draw relative in current scaled units
RDRAWC(XS,YS,IPEN,ITYPE)	same as RDRAW with clipping
RELVEC(X,Y,LINETP)	draw relative to current location
ROTA90(N90)	rotate in 90-deg increments
RXAXIS(LABEL,NCHAR,AXLEN)	draw x relative linear axis
RYAXIS(LABEL,NCHAR,AXLEN)	draw y relative linear axis
SCFTAB	scale tablet (Imlac only)
SETPAG(XAB)	set CalComp plotter end of page
SETTAB(IPREC,LINETP,IUNITS, ITYPE,IDISA)	set INQUIR
SYMBOQ(NCHAR,ITEXT,NSIZE)	draw text in GRAPHICS
TRACON(XARA,YARA,NPTS,NRET,ICHAR)	initiate TRACE mode (Imlac only)
WILDCR(NWILD)	Imlac, GTS/ASC wild-card mode
XAXIS(LABEL,NCHAR,AXLEN)	draw x linear axis
XLOGAX(LABEL,NCHAR,AXLEN)	draw x log axis
YAXIS(LABEL,NCHAR,AXLEN)	draw y linear axis
YLOGAX(LABEL,NCHAR,AXLEN)	draw y log axis

D-2. ROUTINES THAT MUST BE CALLED IN ALPHA MODE ONLY

DELAY(NULLS)	output nulls for delay
ENTGRA	enter GRAPHICS mode
FUNCTIONIWAITQ(IOPCO)	prompt wait and erase
GENGRF(XARA,YARA,NPTS)	general linear graph routine
SCREEN	erase screen while in ALPHA mode
SETPDQ	initialization of GRAPHELP

D-3. ROUTINES THAT ARE MODE INDEPENDENT

AXPREC(IPREC)	fix precision on axis labels
AUTOFR(VAL,IFORM,NFRM,IPREC)	calculate floating format
DELAQ(NSEC)	CPU delay in seconds
FACTOR(XFACT,YFACT)	set overall output size
GETSCA(SF,VLO,IWHO)	get current scale parameters
INIT(XIN,YIN)	set origin coordinates absolutely
INVIRT(XAB,YAB,XS,YS,ITYPE)	convert inches to virtual
LOGQCA(ARRAY,NPTS,AXLEN,INC,IWHO)	log scaling
LOGSCA(AMIN,AMAX,AXLEN,SF, VLO,IWHO)	log scaling
MINMAX(ARRAY,NPTS,INC,AMIN,AMAX)	find min and max value in ARRAY
NUMBRQ(VAL,IPREC,ISIZE)	draw floating point number
PLOT(XIN,YIN,IPEN)	plot absolutely to inches relative to origin
QCALE(AMIN,AMAX,AXLEN,SF,VLO, IWHO)	linear scaling
REGION(NREG)	invoke already stored region

RESCA(ARRAY,NPTS,AXLEN,IWHO, ITYPE)	rescale
RSINCH(XRAS,YRAS,XAB,YAB)	convert rasters to inches
RSVIRT(XRAS,YRAS,XS,YS,ITYPE)	convert raster to virtual
SCALE(ARRAY,NPTS,AXLEN,INC,IWHO)	linear scaling
SERCH(NAME,INDEX)	look up subpicture index
SETWIN(XMN,YMN,XMX,YMX)	set clipping window
STOREG(NREG)	initialize (store) current image space into region
YOURSC(SF,VLO,IWHO)	set your own scale

APPENDIX E.--TWO-DIMENSIONAL AIDS FOR GRAPHELP

The following lists catalogue available routines in GRAPHELP to aid in two-dimensional plotting.

E-1. SCALING ROUTINES

SCALE(ARRAY,NPTS,AXLEN,INC,IWHO)	linear scaling
QCALE(AMIN,AMAX,AXLEN,SF,VLO,IWHO)	linear scaling
LOGQCA(ARRAY,NPTS,AXLEN,INC,IWHO)	log scaling
LOGSCA(AMIN,AMAX,AXLEN,SF,VLO,IWHO)	log scaling
YOURSC(SF,VLO,IWHO)	set user's own scale
GETSCA(SF,VLO,IWHO)	get current scale parameters
RESCA(ARRAY,NPTS,AXLEN,IWHO,ITYPE)	rescale to extreme limits using old scale and ARRAY

E-2. AXIS DRAWING ROUTINES

XAXIS(LABEL,NCHAR,AXLEN)	draw x linear axis
YAXIS(LABEL,NCHAR,AXLEN)	draw y linear axis
RXAXIS(LABEL,NCHAR,AXLEN)	draw x relative linear axis
RYAXIS(LABEL,NCHAR,AXLEN)	draw y relative linear axis
XLOGAX(LABEL,NCHAR,AXLEN)	draw x log axis
YLOGAX(LABEL,NCHAR,AXLEN)	draw y log axis

E-3. DRAWING ROUTINES

DATAQ(XARA,YARA,NPTS,INC,IPEN)	plot linear data
DATLOG(XARA,YARA,NPTS,INC,IPEN,ITYPE)	plot data
DRAW(XS,YS,IPEN,ITYPE)	draw absolute in current scaled units
RDRAW(XS,YS,IPEN,ITYPE)	draw relative in current scaled units
DRAWC(XS,YS,IPEN,ITYPE)	same as DRAW with clipping
RDRAWC(XS,YS,IPEN,ITYPE)	same as RDRAW with clipping

E-4. MISCELLANEOUS

INIT(XIN,YIN)	set origin coordinates absolutely
SETWIN(XMN,YMN,XMX,YMX)	set clipping window
PLOT(XIN,YIN,IPEN)	plot absolute to inches relative to origin
GENGRF(XARA,YARA,NPTS)	general linear graph routine
GRID(XGRD,YGRD,XD,YD,IPEN,IREL)	draw linear grid
NUMBRQ(VAL,IPREC,ISIZE)	draw floating point number
AXPREC(IPREC)	fix precision on axis labels
AUTOFR(VAL,IFORM,NFORM,IPREC)	calculate floating format
MINMAX(ARRAY,NPTS,INC,AMIN,AMAX)	find minimum and maximum values in ARRAY

STOREG(NREG)	initialize (store) current image space into region
REGION(NREG)	invoke already stored region

E-5. PARAMETER DEFINITIONS

AMIN,AMAX	minimum and maximum value of data
ARRAY, XARA, YARA	one-dimensional array of real numbers
NPTS	length of ARRAY
INC	sampling rate of ARRAY
IWHO	axis identification where 0 is x-axis; otherwise y-axis
SF	scale factor in units per inch
VLO	unit value of axis origin
AXLEN	length of axis to perform scaling
LABEL	title for axis (Hollerith array)
NCHAR	number of characters in LABEL
IPREC	number of digits right of decimal point
IPEN	type of line to draw where IPEN < 0 sets new origin and where absolute value of IPEN is IPEN = 1 or 2 solid line IPEN = 3 invisible line IPEN = 4 dotted line IPEN = 5 dashed line IPEN = 6 dot-dashed line any other value assumes solid line
XS,YS	coordinates in scaled logical units
XIN,YIN	coordinate position in inches
XGRD,YGRD	coordinates in inches of upper right grid corner
XD,YD	spacing in inches between grid lines
XMN,YMN,XMX,YMX	boundaries of clipping window
IREL	relative flag IF(IREL.EQ.1), then draw grid with relatives
VAL	real number to output to graphics
IFORM	reserved buffer for format (must be dimensioned at 7)
NFORM	returned parameter of length of format field generated
ISIZE	height of character (0 to 7) (see SYMBOQ)
ITYPE	scaling type currently in effect where 1 x is log, and y is linear 2 x is linear, and y is log 3 x and y are log 4 x and y are linear
NREG	region number from 1 to 7 only

APPENDIX F.--GTS/ASC KEYBOARD DESCRIPTION

F-1. THE ASCII CODE GENERATION

The GTS/ASC keyboard generates the entire 128 character American Standard Code for Information Interchange (ASCII) set. Those characters not displayed on the keyboard can be generated by using the normal control key. The break is held open for as long as the BREAK key is pressed. Each depression of a key outputs one character. Autorepeat is not used except for moving the cross-hair cursor. The GTS/ASC is initially in TTY lock mode which disables all lowercase characters. A REP-SHFT any key, but TAB, toggles this on and off. If the TTY lock is off, then uppercase characters are generated by using the SHFT key. Backspace is generated by the left arrow only when the cross hair is off. Rub out is at the DEL key.

F-2. REPEAT FUNCTIONS

The repeat key has been designated as the local control key for all special GTS/ASC functions. No code is output to the TTY port when the repeat key is pressed with another key.

Transparency inhibits all following input from being processed, except to transfer input from one device to the output of another device. Paper-tape reader input stops whenever an EXIT GRAPHICS or ERASE ALL code is read. Otherwise, it must be turned off from the keyboard. The paper-tape punch when enabled punches all data as they are input to GTS. There is no dump routine of display lists to the paper-tape punch. The Calcomp plotter function is the only device that is not interrupt driven. Consequently, when it is started, there is no way to stop it except by restarting at location 101 at the control console. If the CalComp plotter is started, all input is buffered (up to 128 characters) and then processed on completion of the plot. LOCAL MODE ONLY puts the keyboard in local mode. Input can still be processed from the TTY port.

The reinitialization function REP-O resets all toggles to initial conditions except for the TTY lock. GRAPHICS mode is exited and GTS/ASC is put on line. This is equivalent to restarting at 101.

F-3. CROSS-HAIR CONTROL

The cross hair is not light-pen sensitive at this writing. It can be positioned by using the arrow keys. The screen position is displayed in the upper right corner in decimal raster units. Arrow keys are autorepeat when the cursor is on. Arrow keys are disabled when the cursor is off.

APPENDIX F

F-4. KEYBOARD BUFFER

Up to 54 keyboard inputs can be saved in a special keyboard buffer and output continuously with a single key command. All keyboard characters are legal, including local control functions and special characters. When the buffer is opened (REP-SHFT-TAB), all keyboard input is placed in the buffer until a TAB key is pressed. This closes the buffer. No display is made on the screen while input goes to the buffer. If the buffer overflows, it automatically closes itself. To output the characters put in this buffer, the user presses the TAB key. The buffer is initialized to six blanks.

F-5. LOADER JUMP

The GTS/ASC jumps to the loader at 60 whenever it is in ALPHA mode and it receives an ASCII 2 at the input. Therefore, this can be simulated from the keyboard by going to LOCAL mode and typing CNTRLB or CNTRL-2. The GTS/ASC then transfers control to the loader at 60 and turns off all interrupts.

F-6. THE GTS/ASC KEYBOARD COMMANDS

Key to abbreviations:

REP	repeat key for local control
CNTRL	control key
SHFT	shift key
L.A.	left arrow or backspace
R.A.	right arrow
U.A.	up arrow
D.A.	down arrow
TAB	TAB key
DEL	delete key (rub out)

To toggle the TTY lock, the user keys REP-SHFT any key but TAB.

Generating special characters not on the keyboard:

CNTRL-SHFT A	open square bracket	[
CNTRL-SHFT B	backslash	\
CNTRL-SHFT C	close square bracket	]
CNTRL-SHFT D	up arrow	↑
CNTRL-SHFT E	back arrow(underline)	_
CNTRL-SHFT F	"at" sign	@
CNTRL-SHFT G	open brace	{
CNTRL-SHFT H	vertical bar	|
CNTRL-SHFT I	close brace	}
CNTRL-SHFT J	"similar" sign	~
CNTRL-SHFT K	"grave" sign	`

CNTRL-SHFT TAB	open keyboard TAB buffer
TAB	close keyboard TAB buffer
TAB	output keyboard TAB buffer

Cross-hair cursor control

REP H	enable/disable cross hair
L.A.	move 10 units left
R.A.	move 10 units right
U.A.	move 10 units up
D.A.	move 10 units down
REP L.A.	move 1 unit left
REP R.A.	move 1 unit right
REP U.A.	move 1 unit up
REP D.A.	move 1 unit down

Screen control

REP 1 or REP A	erase ALPHA display
REP 2 or REP B	toggle ALPHA visibility
REP 3 or REP C	toggle ALPHA cursor visibility
REP 4 or REP D	erase GRAPH display
REP 5 or REP E	toggle GRAPH visibility
REP 6 or REP F	toggle GRAPH cursor visibility
REP 7 or REP G	erase screen
REP N	toggle transparency
REP I	input from paper tape reader
REP M	turn off paper tape reader
REP L	toggle paper tape punch
REP 0 (numeral)	hard copy on Calcomp (Versatec)
REP J	LOCAL mode
REP K	ONLINE mode
REP O (letter)	reinitialize GTS/ASC system

DISTRIBUTION

DEFENSE DOCUMENTATION CENTER
CAMERON STATION, BUILDING 5
ALEXANDRIA, VA 22314
ATTN DDC-TCA (12 COPIES)

COMMANDER
USA RSCH & STD GP (EUR)
BOX 65
FPO NEW YORK 09510
ATTN LTC JAMES M. KENNEDY, JR.
CHIEF, PHYSICS & MATH BRANCH

COMMANDER
US ARMY MATERIEL DEVELOPMENT &
READINESS COMMAND
5001 EISENHOWER AVENUE
ALEXANDRIA, VA 22333
ATTN DRXAM-TL, HQ TECH LIBRARY
ATTN PAUL LANGITH

COMMANDER
USA ARMAMENT COMMAND
ROCK ISLAND, IL 61201
ATTN DRSAR-ASF, FUZE DIV
ATTN DRSAR-RDF, SYS DEV DIV - FUZES

COMMANDER
USA MISSILE & MUNITIONS
CENTER & SCHOOL
REDSTONE ARSENAL, AL 35809
ATTN ATSK-CTD-F

US MILITARY ACADEMY
WEST POINT, NY 10996
ATTN LTC CLARENCE SCHEEL
ATTN CAPT RUSSEL WARGA

COMPAID
PO BOX 215
MORRIS PLAINS, NJ 07950
ATTN DR. ROD ALLEN

US DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS
WASHINGTON, DC 20230
ATTN TECHNICAL INFO LIBRARY
ATTN HELEN WOOD, B212 TECH

COMMANDER
US ARMY ELECTRONICS COMMAND
FORT MONMOUTH, NJ 07703
ATTN DRSEL-GG-C, R. DUNN, CAD-E/CAM
ATTN DRSEL-MS, T. DAMES, DEP. DIR.

COMMANDER
PICATINNY ARSENAL
DOVER, NJ 07801
ATTN SARPA-TS-T-S, TECHNICAL LIBRARY
DAVE GRUBSTEIN

COMMANDER
NAVAL SURFACE WEAPONS CENTER
WHITE OAK, MD 20910
ATTN JEFF FRANKLIN

COMMANDER
USA AVIATION SYSTEMS COMMAND
12TH & SPRUCE STREET
PO BOX 209
ST. LOUIS, MO 63166
ATTN DRSA-ERD, DONALD A. BALL

COMMANDER
US ARMY MOBILITY EQUIPMENT
RES & DEV CENTER
FORT BELVOIR, VA 22060
ATTN STSFB-EA, R. MCKECHNIE

COMMANDER
USA MISSILE COMMAND
REDSTONE ARSENAL, AL 35809
ATTN DRSMI-RL, RICHARD EPPS

COMMANDER
USA TANK-AUTOMOTIVE COMMAND
WARREN, MI 48089
ATTN DRST-RHR, JACK PARKS

COMMANDER
ARMY MATERIALS & MECHANICS RES CENTER
WATERTOWN, MA 02172
ATTN DRXMR-TE, RICHARD SHEA

COMMANDER
NATICK LABORATORIES
NATICK, MA 01762
ATTN STSNL-CR, EDWARD JOSEPHSON

COMMANDER
ROCK ISLAND ARSENAL
ROCK ISLAND, IL 61201
ATTN SARRI-L, DR. N. COLEMAN
ATTN SARRI-L, DR. R. BECKETT
ATTN SARRI-LR, W. CASSELL

COMMANDER
EDGEWOOD ARSENAL
EDGEWOOD ARSENAL, MD 21010
ATTN SAREA-CL, LIBRARY

COMMANDER
FRANKFORD ARSENAL
BRIDGE & TACONY STREETS
PHILADELPHIA, PA 19137
ATTN SARFA, LIBRARY

DISTRIBUTION (Cont'd)

COMMANDER
WATERVLIET ARSENAL
WATERVIELT ARSENAL, NY 12189
ATTN SARWV-RDR-C, W. LORENSON

UNIVERSITY OF MICHIGAN
1210 EAST ENGINEERING
ANN ARBOR, MI 48109
ATTN DR. CHACE

IMLAC CORPORATION
SUITE 215
6611 KENILWORTH AVE
RIVERDALE, MD 20840
ATTN JIM FISK

DIRECTOR
DEFENSE ADVANCED RESEARCH
PROJECTS AGENCY
ARCHITECT BLDG
1400 WILSON BLVD
ARLINGTON, VA 22209
ATTN DIR, TECH INFO

DIRECTOR
DEFENSE NUCLEAR AGENCY
WASHINGTON, DC 20305
ATTN SCIENTIFIC INFO DIV

AAI CORPORATION
DEPT 156 E
PO BOX 6767
BALTIMORE, MD 21204
ATTN ED DIETZ

DIRECTOR
NAVAL RESEARCH LABORATORY
WASHINGTON, DC 20390
ATTN TECHNICAL LIBRARY BR

COMMANDER
AF AVIONICS LABORATORY, AFSC
WRIGHT-PATTERSON AFB, OH 45433
ATTN LIBRARY

SOCIAL SECURITY ADMINISTRATION
5681 F. HARPERS FARM RD
COLUMBIA, MD 21044
ATTN MS JACKIE POTTS

ADAGE INC
1079 COMMONWEALTH AVE
BOSTON, MA 02215
ATTN HENRY KERR

CORPORATE INFORMATION SERVICES
UNION OIL CENTER
LOS ANGELES, CA 90017
ATTN PETER HANSEN

SUN OIL COMPANY
1608 WALNUT ST
PHILADELPHIA, PA 19103
ATTN RICHARD W. LINTHICUM/SYSTEMS
DEVELOPMENT

IBM
BOX 390, BLDG 985
RT 55
POUGHKEEPSIE, NY 12602
ATTN CHARLES MENK

UNIVERSITY OF CALIFORNIA
COLLEGE OF ENGINEERING
ELECTRICAL RESEARCH LAB
427 CORY HALL
BERKELEY, CA 94720
ATTN PROF D. O. PEDERSON

HARRY DIAMOND LABORATORIES
ATTN MCGREGOR, THOMAS, COL, COMMANDER/
FLYER, I.N./LANDIS, P.E./
SOMMER, H./OSWALD, R. B.
ATTN CARTER, W.W., DR., TECHNICAL
DIRECTOR/MARCUS, S.M.
ATTN KIMMEL, S., PAO
ATTN CHIEF, 0021
ATTN CHIEF, 0022
ATTN CHIEF, LAB 100
ATTN CHIEF, LAB 200
ATTN CHIEF, LAB 300
ATTN CHIEF, LAB 400
ATTN CHIEF, LAB 500
ATTN CHIEF, LAB 600
ATTN CHIEF, DIV 700
ATTN CHIEF, DIV 800
ATTN CHIEF, LAB 900
ATTN CHIEF, LAB 1000
ATTN RECORD COPY, BR 041
ATTN HDL LIBRARY (3 COPIES)
ATTN CHAIRMAN, EDITORIAL COMMITTEE
ATTN CHIEF, 047
ATTN TECH REPORTS, 013
ATTN PATENT LAW BRANCH, 071
ATTN GIDEP OFFICE, 741
ATTN LANHAM, C., 0021
ATTN COOK, D., 110
ATTN DAMMAN, J., 140
ATTN EDWARDS, A. J., 910
ATTN KIRCHER, K., 0025
ATTN BUSHER, D., 520
ATTN LEE, A., 820
ATTN CHRISTOPHERSON, R., 650
ATTN BIEHL, B., 350
ATTN CHIEF, 0025
ATTN ROSEN, R., 800
ATTN BURKE, E., 722
ATTN CHOY, S., 800 (25 COPIES)